INTERNATIONAL JOURNAL OF MATHEMATICS AND COMPUTERS IN SIMULATION
DOI: 10.46300/9102.2020.14.9 Volume 14, 2020

Efficient method to fault identification, based on grouping components, for industrial processes

Luiza Ocheană, Dan Popescu
Faculty of Automatic Control and Computer Science
University "Politehnica" of Bucharest
Bucharest, Romania
luiza.ocheana@yahoo.com,
dan_popescu_2002@yahoo.com

Luca Ferrarini
Dipartimento di Elettronica e Informazione
Politecnico di Milano
Milan, ITALY
ferrarin@elet.polimi.it

***Abstract* — In this paper we present a novel method for fault identification in Discrete Event Systems, which was developed using the classical diagnoser method. Our goal was to decrease as much as possible the amount of computation that needs to be done to generate a diagnoser and to use it for isolating faults. The proposed method consists in grouping the components of the plant in several types and to generate the plant model, the system model and the diagnoser for the "grouped plant". In this way the size of the automata is reduced. The method is presented and tested on a loading station in a comparative analysis with the method without grouping.**

***Keywords*—diagnosis, fault detection, fault isolation, industrial process.**

I. Introduction

We face a challenge when speak about industrial automation systems: no matter how reliable and intelligent the control system is, we may still encounter different kind of faults. Not only that we cannot find perfect equipments, but their behavior can leading to faults. Because of financial and even human losses, it is necessary to detect and isolate the faults as soon as possible. In order to detect and isolate the failures, appropriate methods of diagnosis are needed. For this purpose all the available details about the process and its functionality are taken into account.

Failure detection in dynamic systems was investigated over the last decades [11] and it still is. Even since 1995 the detection and isolation of faults in discrete event systems has received a lot of attention [9]. Later this work was extended in many directions. In [1] the authors described a method for computing the observation likelihood of a stochastic automaton and use this method to find the most-likely stochastic automaton. In [10] the authors developed sequential window diagnosers (SWDs) utilizing the notions of state probability vector and stochastic diagnoser probability transition matrices. Ribot et al. presented in [7] a formal characterization of the diagnostic and prognostic problems in order to support the maintenance of a complex system. In [5] the authors proposed a model-based approach to passive online fault diagnosis for timed systems, describing the system to be diagnosed as a network of communicating timed automata. In [2] the authors described a whole methodology for the design of a real-time diagnostic system, from the specification to implementation, along with a complete testing on a real industrial automated system. A novel practical algorithm for real-time diagnosis suitable for automated devices (TiDiaM) is presented in [3]. Rincon [8] presented the study of the detection and diagnosis of multiple faults in a Gas Turbine using principal component analysis and structured residuals method.

One challenge in large industrial systems is that we may have multiple faults at one moment of time, and the personnel do not have enough time and expertise to fix all of them. One approach that can be used for large industrial systems is to divide the system in small and independent parts. Unfortunately, this method may skip some of the interactions between the components.

Another approach focuses on the diagnosis of only one type of fault using one diagnoser for each fault. Pencolé et al. proposed to analyse the system in order to detect a subsystem that is sufficient for diagnosing a particular type of fault [6]. The classical diagnoser method will become very difficult in large and complex systems, mainly due to the large size of the sets of descriptive parameters. Large sets lead to the necessity of large computing and storage capabilities. This paper presents a method for fault detection that was born from the need to reduce the amount of computation that has to be done in order to isolate a fault as soon as possible.

The rest of the paper is organized as follows: in Section II all the necessary notations and definitions are introduced in order to establish the basis for Section III which introduces the new method for fault detection, based on grouping criteria. Section IV describes an example of implementation on a loading station. Section V concludes the paper.

II. The Proposed Method

The method we will present is based on the basic rules of building the diagnoser automaton described in [4] and [9]. The method involves, in the first phase, partitioning the process into disjunctive groups and in the second phase, detecting the fault that occur in the operation of the process. By this method, we obtain an important reduction in the required number of states

used to represent the process and the attached controller, which accelerates the process of tracking and identifying errors.

III.1. Grouping the components

The efficiency of the method is strongly influenced by the chosen grouping criteria depending on the application we consider. Within an industrial plant, we can group the automation equipment into several types, according to the following criteria:

A. Types of components

The components of a facility/plant can be grouped in two main categories, each of them consisting in several types:

- execution components: pneumatic valves, electric valves, pumps, compressors, engines, and so on;
- measurement components - sensors: pressure, temperature, flow, level, weighing, position and so on.

By grouping all or some components of the same type (for example pneumatic valves), the corresponding model could be described by an automaton having the same number of states and events.

B. The operation of the components

Depending on the operation of the plant as a whole and the control logic, each type can be divided into several categories. For example, a category can consist of valves that act similar in the operation (meaning that will be either closed or opened at the same time) or work alternatively (will not be all closed or opened at the same time). This criterion is highly dependent on the particularities of the plant and the control sequences.

C. Response time

The response time of the components is not a criterion for the method described above. However, if we want to supplement it with time used in TiDiaM, it becomes a very important criterion to consider.

III.2. Method

Suppose that the plant has n components:

$$\Pi = \{P_1, P_2, \dots, P_n\} \tag{1}$$

and the plant model P is a parallel composition of its components:

$$P = P_1 || P_2 || \dots || P_n \tag{2}$$

where P_i are automata described by:

$$P_i = \left(X_{pi}, \Sigma_{pi}, \delta_{pi}, x_{0pi}, x_{fpi}\right), i \epsilon \{1,2, \dots, n\} \tag{3}$$

Suppose we group the components in several types, according to one or more criteria presented above. We define the following partition of Π:

$$\Pi = \Pi_1 \cup \Pi_2 \cup \dots \cup \Pi_s, s < n \tag{4}$$

where:

$$\Pi_h = \{P_{h1}, P_{h2}, \dots, P_{hk}\} - type\ h \tag{5}$$

and:

$$\begin{cases} \Pi_h \subset \Pi, h \in \{1,2, \dots, s\}, \Pi_i \cap \Pi_j = \emptyset\ \forall\ i \neq j \\ i, j \in \{1,2, \dots, s\} \end{cases} \tag{6}$$

For each group (type h):

$$\Pi_h = \{P_{h1}, P_{h2}, \dots, P_{hk}\}, h \in \{1,2, \dots, s\} \tag{7}$$

we consider an automaton:

$$G_h = \left(X_{Gh,} \Sigma_{Gh}, \delta_{Gh}, x_{0Gh}, x_{fGh}\right) \tag{8}$$

where the elements of the sets $X_{Gh,}$ and Σ_{Gh} are dependent on the corresponding elements of process components of Π_h. So, if:

$$\begin{cases} P_{hi} = \left(X_{hi}, \Sigma_{hi}, \delta_{hi}, x_{0hi}, x_{fhi}\right) \\ i \in \{1,2, \dots, k\}, h \in \{1,2, \dots, s\} \end{cases} \tag{9}$$

and:

$$\begin{cases} X_{hi} = \{x_{hi}^1, x_{hi}^2, \dots, x_{hi}^a\} \\ \Sigma_{hi} = \{\sigma_{hi}^1, \sigma_{hi}^2, \dots, \sigma_{hi}^b\} \\ i \in \{1,2, \dots, k\}, h \in \{1,2, \dots, s\} \end{cases} \tag{10}$$

then:

$$\begin{cases} X_{Gh} = \{x_{Gh}^1, x_{Gh}^2, \dots, x_{Gh}^a\} \\ \Sigma_{Gh} = \{\sigma_{Gh}^1, \sigma_{Gh}^2, \dots, \sigma_{Gh}^b\}, h \in \{1,2, \dots, s\} \end{cases} \tag{11}$$

are defined as:

$$\begin{cases} x_{Gh}^i = f_{X_{Gh}}(x_{h1}^i, x_{h2}^i, \dots, x_{hk}^i) \\ \sigma_{Gh}^i = f_{\Sigma_{Gh}}(\sigma_{h1}^i, \sigma_{h2}^i, \dots, \sigma_{hk}^i) \\ h \in \{1,2, \dots, s\}, i \in \{1,2, \dots, k\} \end{cases} \tag{12}$$

The set of group automata is defined as:

$$\Pi_G = \{G_1, G_2, \dots, G_s\} \tag{13}$$

For $f_{X_{Gh}}$ and $f_{\Sigma_{Gh}}$ we consider simple functions implemented by the following architecture (Fig. 1):

Based on this grouping, we construct a group-based model of the plant *G* as a parallel composition of group models:

$$\begin{cases} G = G_1||G_2||\dots||G_s \\ G = (X_G, \Sigma_G, \delta_G, x_{0G}, x_{fG}) \end{cases} \quad (14)$$

It is obvious that the dimension (the number of states and the number of events) of G is lower that the dimension of P.

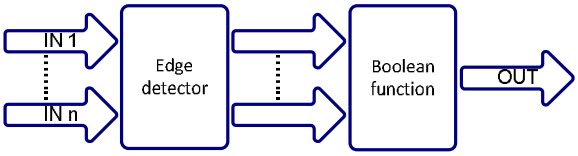

Fig. 1. Architecture of a typical function of group component

The controller for P is defined as:

$$C_P = (X_{CP}, \Sigma_{CP}, \delta_{CP}, x_{0CP}, x_{fCP}) \quad (15)$$

$$\begin{cases} X_{CP} = X_{CP1} \cup X_{CP2} \cup \dots \cup X_{CPn} \\ \Sigma_{CP} = \Sigma_{Cp1} \cup \Sigma_{Cp2} \cup \dots \cup \Sigma_{Cpn} \\ \delta_{CP}: X_{CP} X \Sigma_{CP} \to X_{CP} \end{cases} \quad (16)$$

where X_{CPi} are the states of the controller corresponding to P_i, Σ_{Cpi} – is the set of control events corresponding to P_i, δ_{CP} - is the transition function between the states, x_{0CP} - represents the initial (start) state and x_{fCP} - represents the final state.

We construct the group-based controller model:

$$C_G = (X_{CG}, \Sigma_{CG}, \delta_{CG}, x_{0CG}, x_{fCG}) \quad (17)$$

where:

$$\begin{cases} X_{CG} = X_{CG1} \cup X_{CG1} \cup \dots \cup X_{CGs} \\ \Sigma_{CG} = \Sigma_{CG1} \cup \Sigma_{CG1} \cup \dots \cup \Sigma_{CGs} \\ \delta_{CG}: X_{CG} X \Sigma_{CG} \to X_{CG} \end{cases} \quad (18)$$

and X_{CGi} – are the states of the controller corresponding to G_i, Σ_{CGi} – is the set of control events corresponding to G_i, δ_{CG} - is the transition function between the states, x_{0CG} - represents the initial (start) state, x_{fCG} - represents the final state.

Because of the way it was build, the group controller has fewer states and events that the initial one.

We construct the system model for P:

$$S_P = P||C_P, S_P := (X_{SP}, \Sigma_{SP}, \delta_{SP}, x_{0SP}, x_{fSP}) \quad (19)$$

and the group system model for G:

$$S_G = G||C_G, S_G := (X_{SG}, \Sigma_{SG}, \delta_{SG}, x_{0SG}, x_{fSG}) \quad (20)$$

Then the number of states and the number of events of S_G is lower than the ones of S_P.

III.3. Fault detection

Finally, we compute the diagnoser, according to the classical method.

We define the failure partition for P:

$$\begin{cases} \Delta f_P = F_1 \cup F_2 \cup \dots \cup F_n \\ F_i - \text{the set of failure for the component } P_i \end{cases} \quad (21)$$

We group F_i in accordance with (5):

$$\begin{cases} \Delta f_P = F_{\Pi_1} \cup F_{\Pi_2} \cup \dots \cup F_{\Pi_s} \\ F_{\Pi_i} - \text{the set of failure for the component } \Pi_i \end{cases} \quad (22)$$

$$\begin{cases} F_{\Pi_1} = F_{11} \cup F_{12} \cup \dots \cup F_{1i} \\ F_{\Pi_2} = F_{21} \cup F_{22} \cup \dots \cup F_{2j} \\ \dots \\ F_{\Pi_s} = F_{s1} \cup F_{s2} \cup \dots \cup F_{sk} \end{cases} \quad (23)$$

where:

$$\begin{cases} F_{1a} = \{F_{1a}^1, F_{1a}^2, \dots, F_{1a}^i\}, a \in \{1,2,\dots,i\} \\ F_{2b} = \{F_{2b}^1, F_{2b}^2, \dots, F_{2b}^j\}, b \in \{1,2,\dots,j\} \\ \dots \\ F_{sc} = \{F_{sc}^1, F_{sc}^2, \dots, F_{sc}^k\}, c \in \{1,2,\dots,k\} \end{cases} \quad (24)$$

We can now define the failure partition for G as:

$$\begin{cases} \Delta f_G = F_{G1} \cup F_{G2} \cup \dots \cup F_{Gs} \\ F_{Gi} - \text{set of failure labels for } G_i \end{cases} \quad (25)$$

where:

$$\begin{cases} F_{G1} = \{F_{G1}^1, F_{G1}^2, \dots, F_{G1}^i\} \\ F_{G2} = \{F_{G2}^1, F_{G2}^2, \dots, F_{G2}^j\} \\ \dots \\ F_{Gs} = \{F_{Gs}^1, F_{Gs}^2, \dots, F_{Gs}^k\} \end{cases} \quad (26)$$

$$\begin{cases} F_{G1}^{a} = f_{G1}(F_{11}^{a}, F_{12}^{a}, \dots, F_{1i}^{a}), a \in \{1,2,\dots,i\} \\ F_{G2}^{b} = f_{G2}(F_{21}^{b}, F_{22}^{b}, \dots, F_{2j}^{b}), b \in \{1,2,\dots,j\} \\ \dots \\ F_{Gs}^{c} = f_{Gs}(F_{s1}^{c}, F_{s2}^{c}, \dots, F_{sk}^{c}), c \in \{1,2,\dots,k\} \end{cases} \quad (27)$$

We can now compute the diagnosers: one for the classical model of the system S and the failure partition Δf_P:

$$D_P = (X_{DP}, \Sigma_{DP}, \delta_{DP}, x_{0DP}, x_{fDP}) \quad (28)$$

and one using the group system model S_G and the failure partition Δf_G:

$$D_G = (X_{DG}, \Sigma_{DG}, \delta_{DG}, x_{0DG}, x_{fDG}) \quad (29)$$

It is now obvious that the dimension of Δf_G is lower than the dimension of Δf_P. Therefore:

$$|X_{DG}| < |X_{DP}|, |\Sigma_{DG}| < |\Sigma_{DP}| \quad (30)$$

Because all the automata involved in the generation of D_G are smaller than the ones used for D_P, the diagoser D_G will also have fewer states and fewer events than the classical one.

When a failure occurs, D_G will isolate the fault and will estimate it belongs to F_{Gi}, meaning a fault at a component from P_i (of type $i \in \{1,2,\dots,s\}$).

An automation system usually includes an events history component that keeps the record of all the observable events that occurred. Consider we store the observable events recorded until the fault occurred, in an array, we can search for the exact component that has a fault by searching the event from Σ_{pi} that appeared.

The method we have presented is highly dependent on the degree of grouping; therefore, on the particularities of the plant and the control sequences.

This is the reason why the most important disadvantage of the method we have presented above is that there may be situations (plants) in which the method will not bring major improvements or will not bring any improvements at all to the classical method.

This can happen especially in small processes with few components, or composed of several sub-processes between which the interaction is limited.

But for small processes the conventional method is not a challenge, since the amount of information and computing is not high.

For processes that consist in several sub-processes, because of the limited interaction, it is easy to build separated diagnosers for each sub-process, returning in the situation above.

This is why the criteria we have presented do not limit the application of the method to a certain category of processes (plants), but represent the rules for defining the generalized plant model.

III. Example of Implementing the Method

In order to demonstrate the effectiveness of the method we presented above, we exemplified it on a small loading station used to load fuel from fuel trucks into fuel tanks (Fig. 2). The station is composed of two pumps, three valves and the piping between. The components and the operations of the considered system are the following:

- One valve on the inlet pipe;
- Two fuel pumps;
- Two valves on each line to the tanks.

All valves are of the same type, dimensions and are provided with limit switches for monitoring their position in the control system.

The operation of the loading station:

The initial state is the same as the safe state of the plant and is described as following: pumps stopped, valves closed.

The start-up procedure is composed of the following sequence:

- Open XV1;
- Choose the tank for loading;
- Open the corresponding valve (either XV1 or XV2);
- Start P1 or P2 (if P1 is in maintenance).

The station will be shut down in the following situations:

- Loading finished;
- Automatic alarm;
- Manual alarm.

The shut down procedure consists in stopping the pumps and closing the valves.

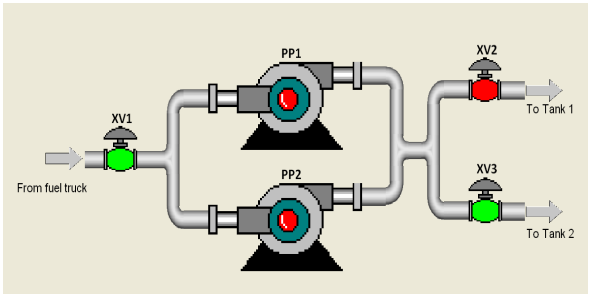

Fig. 2. Loading station

Using the classical diagnoser approach, we would construct the automaton for every component in order to achieve the plant model as parallel composition. Next, we would construct the control model and the system model. For our examples we used DESUMA/UMDES **Error! Reference source not found.** software developed from the University of Michigan, which

permits the generation of the automata of the system (plant and control models) and the diagnoser automaton.

For the above example we define the plant Π having 5 components:

$$\Pi = \{P_1, P_2, P_3, P_4, P_5\} \tag{31}$$

Where:

$$\begin{cases} P_1 - \ is\ the\ automaton\ for\ XV1 - \ Fig.4 \\ P_2 - \ is\ the\ automaton\ for\ PP1 - \ Fig.5 \\ P_3 - \ is\ the\ automaton\ for\ PP2 - Fig.5 \\ P_4 - \ is\ the\ automaton\ for\ XV2 - \ Fig.6 \\ P_5 - \ is\ the\ automaton\ for\ XV3 - \ Fig.7 \end{cases} \tag{32}$$

According to (9) we have:

$$\begin{cases} P_1 = (X_1, \Sigma_1, \delta_1, x_{01}, x_{f1}) \\ P_2 = (X_2, \Sigma_2, \delta_2, x_{02}, x_{f2}) \\ P_3 = (X_3, \Sigma_3, \delta_3, x_{03}, x_{f3}) \\ P_4 = (X_4, \Sigma_4, \delta_4, x_{04}, x_{f4}) \\ P_5 = (X_5, \Sigma_5, \delta_5, x_{05}, x_{f5}) \end{cases} \tag{33}$$

and:

$$\begin{gathered} X_1 = \{XV1\_O, XV1\_C, XV1\_SO, XV1\_SC, \\ XV1\_C: so1\_XV1, XV1\_O: sc2\_XV1\} \\ \Sigma_1 = \{ZS1\_C \to ZS1\_O, ZS1\_O \to ZS1\_C, \\ (c\_XV1, ZS1\_C), (c\_XV1, ZS1\_O), (o\_XV1, ZS1\_C), \\ (o\_XV1, ZS1\_O), sc1\_XV1, sc2\_XV1, \\ so1\_XV1, so2\_XV1\} \\ X_2 = \{P1\_STARTED, P1\_STOPPED\} \\ \Sigma_2 = \{stop\_p1, start\_p1\} \\ X_3 = \{P2\_STARTED, P2\_STOPPED\} \\ \Sigma_3 = \{stop\_p1, start\_p1\} \\ X_4 = \{XV2\_O, XV2\_C, XV2\_SO, XV2\_SC, \\ XV2\_C: so1\_XV2, XV2\_O: sc2\_XV2\} \\ \Sigma_4 = \{ ZS2\_C \to ZS2\_O, ZS2\_O \to ZS2\_C, \\ (c\_XV2, ZS2\_C), (c\_XV2, ZS2\_O), (o\_XV2, ZS2\_C), \\ (o\_XV2, ZS2\_C), sc1\_XV2, sc2\_XV2, \\ so1\_XV2, so2\_XV2\} \\ X_5 = \{XV3\_O, XV3\_C, XV3\_SO, XV3\_SC, \\ XV3\_C: so1\_XV3, XV3\_O: sc2\_XV3\} \\ \Sigma_5 = \{ZS3\_C \to ZS3\_O, ZS3\_O \to ZS3\_C, \\ (c\_XV3, ZS3\_C), (c\_XV3, ZS3\_O), (o\_XV3, ZS3\_C), \\ (o\_XV3, ZS3\_C), sc1\_XV3, sc2\_XV3, \\ so1\_XV3, so2\_XV3\} \end{gathered} \tag{34}$$

We consider that the pumps are ideal, they do not have faults.

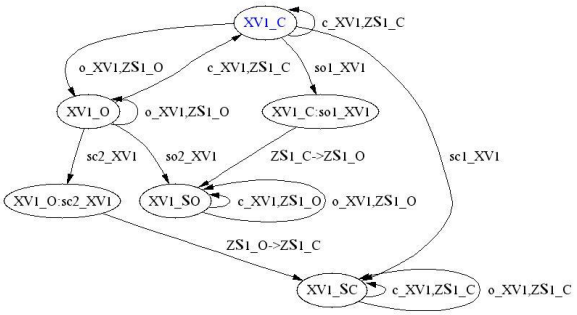

Fig. 3. The automaton for XV1

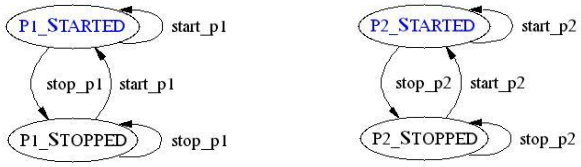

Fig. 4. The automata for PP1 and PP2

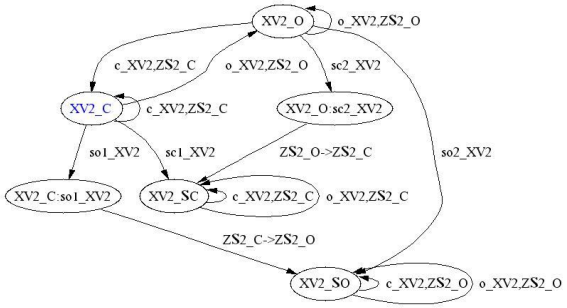

Fig. 5. The automaton for XV2

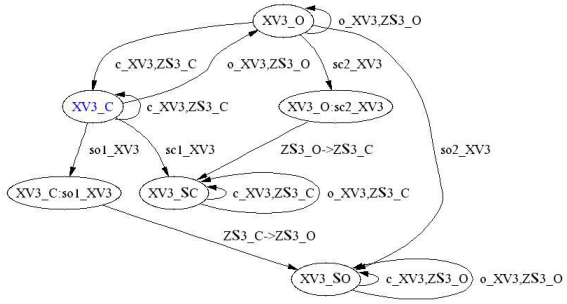

Fig. 6. The automaton for XV3

The plant model is computed as the parallel composition of its components according to (2):

$$P = P_1||P_2||P_3||P_4||P_5 \quad (35)$$

We define the control model C_P and we compute the system model for P according to (19).

Now we can summarize: following the classical method we obtain:

- The plant model P having 864 states;
- The control model C_P having 16 states;
- The system model S_P having 8640 states.

We define the failure partition:

$$\Delta f_P = \{F_1, F_2, F_3, F_4, F_5\} \quad (36)$$

$$\begin{cases} F_1 = \{so1\_xv1, sc1\_xv1, so2\_xv1, sc2\_xv1\} \\ F_2 = \emptyset \\ F_3 = \emptyset \\ F_4 = \{so1\_xv2, sc1\_xv2, so2\_xv2, sc2\_xv2\} \\ F_5 = \{so1\_xv3, sc1\_xv3, so2\_xv3, sc2\_xv3\} \end{cases} \quad (37)$$

We compute the diagnoser D_P with 10 233 states.

Using the method we have described in section III.2, with the following hypothesis:

- PP1 and PP2 will never be both started at the same time;
- XV2 and XV3 will never be both opened at the same time;
- The pumps are ideal, they do not have faults,

it is possible to group the components in 3 types of components using OR function:

- 1st group: XV1;
- 2nd group: PP1, PP2;
- 3rd group: XV2, XV3.

We define the following partition of Π:

$$\Pi = \Pi_1 \cup \Pi_2 \cup \Pi_3 \quad (38)$$

where:

$$\begin{cases} \Pi_1 = \{P_1\} - type\ 1 \\ \Pi_2 = \{P_2, P_3\} - type\ 2 \\ \Pi_3 = \{P_4, P_5\} - type\ 3 \end{cases} \quad (39)$$

We construct the group automata G_1, G_2, G_3:

$$\begin{cases} G_1 = (X_{G1}, \Sigma_{G1}, \delta_{G1}, x_{0G1}, x_{fG1}) - \text{Fig. 7} \\ G_2 = (X_{G2}, \Sigma_{G2}, \delta_{G2}, x_{0G2}, x_{fG1}) - \text{Fig. 8} \\ G_3 = (X_{G3}, \Sigma_{G3}, \delta_{G3}, x_{0G3}, x_{fG3}) - \text{Fig. 9} \end{cases} \quad (40)$$

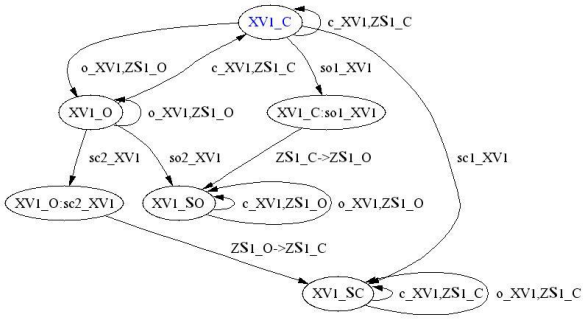

Fig. 7. The automaton for G_1

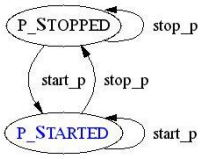

Fig. 8. The automaton for G_2

where:

$$\begin{cases} X_{G1} = \{XV1\_O, XV1\_C, XV1\_SO, XV1\_SC, \\ XV1\_C{:}so1\_XV1, XV1\_O{:}sc2\_XV1\} \\ \Sigma_{G1} = \{ ZS1\_C \to ZS1\_O, ZS1\_O \to ZS1\_C, \\ (c\_XV1, ZS1\_C), (c\_XV1, ZS1\_O), \\ (o\_XV1, ZS1\_C), (o\_XV1, ZS1\_C), \\ sc1\_XV1, sc2\_XV1, so1\_XV1, so2\_XV1\} \\ X_{G2} = \{P\_STOPPED, P\_STARTED\} \\ \Sigma_{G2} = \{start\_p, stop\_p\} \\ X_{G3} = \{XV23\_O, XV23\_C, XV23\_SO, XV23\_SC, \\ XV23\_C{:}so1\_XV23, XV23\_O{:}sc2\_XV23\} \\ \Sigma_{G3} = \{ZS23\_C \to ZS23\_O, ZS23\_O \to ZS23\_C, \\ (c\_XV23, ZS23\_C), (c\_XV23, ZS23\_O), \\ (o\_XV23, ZS23\_C), (o\_XV23, ZS23\_C), \\ sc1\_XV23, sc2\_XV23, so1\_XV23, so2\_XV23\} \end{cases} \quad (41)$$

and:

$$\begin{cases} P\_STOPPED = f_{XG2}(P1\_STOPPED, P2\_STOPPED) \\ P\_STARTED = f_{XG2}(P1\_STARTED, P2\_STARTED) \\ XV23\_O = f_{XG3}(XV2\_O, XV3\_O) \\ XV23\_C = f_{XG3}(XV2\_C, XV3\_C) \\ XV23\_SO = f_{XG3}(XV2\_SO, XV3\_SO) \\ XV23\_SC = f_{XG3}(XV2\_SC, XV3\_SC) \\ XV23\_C{:}so1\_XV23 = f_{XG3}(XV2\_C{:}so1\_XV2, XV3\_C{:}so1\_XV3) \\ XV23\_O{:}sc2\_XV23 = f_{XG3}(XV2\_O{:}sc2\_XV2, XV3\_O{:}sc2\_XV3) \end{cases} \quad (42)$$

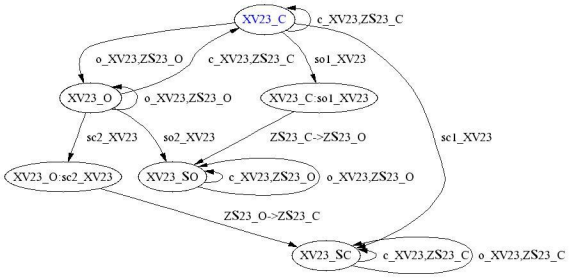

Fig. 9. The automaton for G_3

$$\begin{cases} start\_p = f_{\Sigma_{G2}}(start\_p1, start\_p2) \\ stop\_p = f_{\Sigma_{G2}}(stop\_p1, stop\_p2) \\ ZS23\_C \to ZS23\_O = f_{\Sigma_{G3}}(ZS2\_C \to ZS2\_O, ZS3\_C \to ZS3\_O) \\ ZS23\_O \to ZS23\_C = f_{\Sigma_{G3}}(ZS2\_O \to ZS2\_C, ZS3\_O \to ZS3\_C) \\ (c\_XV23, ZS23\_C) = f_{\Sigma_{G3}}((c\_XV2, ZS2\_C), (c\_XV3, ZS3\_C)) \\ (c\_XV23, ZS23\_O) = f_{\Sigma_{G3}}((c\_XV2, ZS2\_O), (c\_XV3, ZS3\_O)) \\ (o\_XV23, ZS23\_C) = f_{\Sigma_{G3}}((o\_XV2, ZS2\_C), (o\_XV3, ZS3\_C)) \\ (o\_XV23, ZS23\_C) = f_{\Sigma_{G3}}((o\_XV2, ZS2\_C), (o\_XV3, ZS3\_C)) \\ sc1\_XV23 = f_{\Sigma_{G3}}(sc1\_XV2, sc1\_XV3) \\ sc2\_XV23 = f_{\Sigma_{G3}}(sc2\_XV2, sc2\_XV3) \\ so1\_XV23 = f_{\Sigma_{G3}}(so1\_XV2, so1\_XV3) \\ so2\_XV23 = f_{\Sigma_{G3}}(so2\_XV2, so2\_XV3) \end{cases} \quad (43)$$

For $f_{X_{G2}}$, $f_{\Sigma_{G2}}$, $f_{X_{G3}}$ and $f_{\Sigma_{G3}}$ we consider the architecture presented in Fig. 1, for a number of 2 inputs.

It is now possible to compute the group-based model of the plant G as a parallel composition of group models according to (14):

$$G = G_1 || G_2 || G_3 \quad (44)$$

We construct the group-based controller model (Fig. 10) and system model based on (17) and (20).

In this case:

- The plant model has 72 states;
- The control model has 6 states;
- The system model has 324 states.

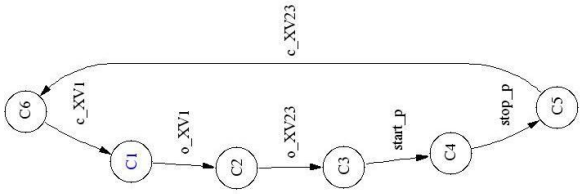

Fig. 10. The group-based controller model

We group F_i defined in (42):

$$\Delta f_P = F_{\Pi_1} \cup F_{\Pi_2} \cup F_{\Pi_3} \quad (45)$$

where:

$$\begin{cases} F_{\Pi_1} = F_1 \\ F_{\Pi_2} = F_2 \cup F_3 \\ F_{\Pi_3} = F_4 \cup F_5 \end{cases} \quad (46)$$

We define Δf_G:

$$\Delta f_G = F_{G1} \cup F_{G2} \cup F_{G3} \quad (47)$$

where:

$$\begin{cases} F_{G1} = \{so1\_xv1, so2\_xv1, sc1\_xv1, sc2\_xv1\} \\ F_{G2} = \emptyset \\ F_{G3} = \{so1\_xv23, so2\_xv23, sc1\_xv23, sc2\_xv23\} \end{cases} \quad (48)$$

and:

$$\begin{cases} so1\_xv23 = f_G(so1\_xv2, so1\_xv3) \\ so2\_xv23 = f_G(so2\_xv2, so2\_xv3) \\ sc1\_xv23 = f_G(sc1\_xv2, sc1\_xv3) \\ sc2\_xv23 = f_G(sc2\_xv2, sc2\_xv3) \end{cases} \quad (49)$$

We compute the diagnoser D_G for the group-based system model and the failure partition Δf_G having 429 states, unlike the initial one with 10 233 states.

D_G will act as a classical diagnoser for the group-based system model, meaning it will isolate the group of faults (F_{Gi}). After isolating the group we use the algorithm presented in **Error! Reference source not found.** to find the component within the group that has a fault.

For example, if the fault $so1\_xv2$ occurs, D_G will isolate F_{G3}, meaning we have a fault either in XV2 or XV3. Next, we compare the observable events that occurred (recorded by the monitoring and control system of the plant) with the sets of events Σ_4 and Σ_5. We will find that one of the latest observable event belongs to Σ_4 meaning we have a failure at XV2.

IV. Conclusions

In this paper we have presented a new method based on grouping components for detecting and isolating faults in industrial automated systems. The method is based on the classical diagnoser approach, but it significantly reduces the number of states in the plant model, the dimension of the set of failure labels and consequently the number of states of the diagnoser. This is very important in large applications where the computing and storage capacity should be high for the classical method.

The applicability and efficiency of the method are process dependent, meaning that it depends on the structure and functioning of the automated process how much the number of states is reduced and how effective the method can be.

References

[1] E. Athanasopoulou, L. Lingxi, and C.N. Hadjicostis (2006). Probabilistic failure diagnosis in finite state machines under unreliable observations, In Proc. of 2006 8th International Workshop on Discrete Event Systems, pp. 301 – 306;

[2] L. Ferrarini, M. Allevi, and A. Dede (2011). A Methodology for Fault Isolation and Identification in Automated Equipments, 9th IEEE International Conference on Industrial Informatics, pp. 157 – 162;

[3] L. Ferrarini, M. Allevi, and A. Dede (2011). Implementation and testing of an online fault isolation methodology in a real industrial scenario, 3rd International Workshop on Dependable Control of Discrete Systems (DCDS), pp. 13 – 18;

[4] L. Ferrarini, M. Allevi, and A. Dede (2011). A real-time algorithm for fault identification in machining centres. IFAC2011, pp. 5201-5206;

[5] E. Gascard and Z. Simeu-Abazi (2011). Automatic Construction of Diagnoser for Complex Discrete Event Systems, 3rd International workshop on Dependable Control of Discrete systems, pp. 90 - 95;

[6] Y. Pencolé, D. Kamenetsky and A. Schumann (2006). Towards Low-Cost Fault Diagnosis in Large Component-Based Systems, 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, pp. 1473-1478;

[7] P. Ribot, Y. Pencol´e and M. Combacau (2009). Diagnosis and prognosis for the maintenance of complex systems, IEEE International Conference on Systems, Man and Cybernetics, pp. 4146 – 4151;

[8] A. Rincon (2012). Multiple fault detection and diagnosis in a Gas Turbine using principal component analysis and structured residuals. 20th Mediterranean Conference on Control & Automation (MED), pp. 91 - 97;

[9] M. Sampath, R. Sengupta, S. Lafortune, K. Sinaamohideen and D.Teneketzis (1995). Diagnosability of discrete event systems, IEEE Transactions on Automatic Control, 40(9), pp. 1555–1575;

[10] L. Wen-Chiao, H. E. Garcia, D. Thorsley, and T. Yoo (2009). Sequential Window Diagnoser for Discrete - Event Systems Under Unreliable Observations, Allerton'09 Proceedings of the 47th annual Allerton conference on Communication, control, and computing, pp. 668-675;

[11] A. S. Willsky (1976). A Survey of Design Methods for failure Detection in Dynamic Systems (vol. 12) Automatica, 12, pp. 601-611;