An Architecture for Real Time Diagnosis of Gas Turbines

Luis de Jesús González–Noriega and Pablo H. Ibargüengoytia

Instituto de Investigaciones Eléctricas
Av. Reforma 113, Palmira
Temixco, Mor., 62490, México
ljgon@hotmail.com / pibar@iie.org.mx

Abstract. Real time diagnosis systems are becoming an important requirement in these days given the complexity of industrial systems. Some diagnosis systems obtain data from a working process and later, off line, they run diagnosis in order to explain abnormal behavior. This article presents an architecture for real time diagnosis based on probabilistic reasoning. Probabilistic reasoning utilizes a model of the system that expresses the probabilistic relationship between the main variables of a process. Thus, the values of some variables are utilized as evidence and the propagation provides an inferred value of other variables and an abnormal condition is detected. Next, an isolation phase is executed in order to find the root cause of the abnormal behavior.
This article presents the design of an architecture that performs real time diagnosis of gas turbines of combined cycle power plants. The architecture was designed utilizing some of the classes of the Spanish *elvira* project as a double experiment: (i) to test a general purpose, probabilistic reasoning package elvira in a real application in real time and (ii) to test a previously developed theory for real time diagnosis in a gas turbine.

1 Introduction

Given the high costs and the difficulties for building new electric generation plants, the current trend consists in increasing the performance, availability and reliability of the actual installations. The performance refers to the amount of mega watts that can be generated with a unit of fuel. The availability refers to the hours that the central stops generating, and the reliability refers to the probability of counting with the different equipment in the plant.

Diagnosis is the technique utilized in several fields devoted to find faults, to explain abnormal behavior or to detect a faulty component in a system. Sometimes, engineers acquire data from a working process in order to analyze it off line, and detect the faulty component or the cause of the abnormal behavior. Other diagnosis systems run in line, i.e., they collect data and reason about the responses of the system. Some diagnosis systems predict the occurrence of a failure while others only explain the causes once that the fault has been propagated.

The diagnosis architecture presented in this paper is part of a larger system formed by a monitor, an optimizer, a diagnoser and an intelligent planning module. When the monitor detects that the process works normally, it runs the optimizer in order to increase the performance of the process, e.g., the generation of more mega watts with less fuel. On the opposite, when the diagnoser detects an abnormal behavior, it identifies the faulty component and starts the intelligent planning that generates advices to the operator in order to return the plant to its normal state. This optimizer and diagnosis system are devoted to enhance the performance and availability indices.

The diagnosis system utilizes probabilistic reasoning for the detection and isolation of faults. It is based on the reasoning that an experimented operator carries out when some sensors of the plant present abnormal readings given the rest of signals. The operator may infer a fault in a sensor given the readings of the related variables. The first step is to acquire a probabilistic model of the process that relates the variables, i.e., a Bayesian network. Then one by one, the variables are predicted through probabilistic propagation and compared with the real value. If significant deviation is presented, then an apparent fault is detected. The isolation of the fault is carried out using a special property of the Bayesian networks called the Markov blanket.

The diagnosis architecture is based on the *elvira project*[1]. The input for the system is a sample with data of one execution of the process without any failure. With the data, the learning module of elvira generates a Bayesian network representing the probabilistic relationship between all the variables considered. Once a model is established, the system runs in real time, i.e., it reads data from the process and detects faulty components. The output of the diagnosis is a vector with the probability of failure of all the variables considered.

This paper is organized as follows. Section 2 explains a previously developed theory for probabilistic validation of important variables. Next, section 3 explains the proposed architecture, and section 4 briefly describes the application domain and the experiments carried out with the architecture and a gas turbine simulator in laboratory. Finally, section 5 concludes the paper and proposes future work in this area.

2 Probabilistic Diagnosis Model

The probabilistic diagnosis model requires the construction of a Bayesian network relating all the variables in the system being diagnosed. For example, Fig. 1 shows a probabilistic model of a process where variables g and *a cause* variable t, and this variable together with p affect the behavior of variable m.

The diagnosis consists of two operations: (i) *basic validation* and (ii) *isolation*. The former operation detects the presence of a fault while the later isolates the faulty element. This corresponds to the FDI (Fault Detection and Isolation) technique utilized in industry. These operations are made cyclically with all the

[1] information about this project can be consulted in http://www.ia.uned.es/ẽlvira.

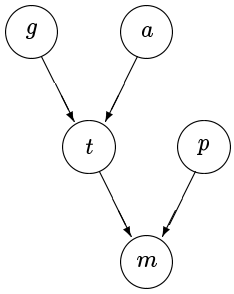

Fig. 1. A Bayesian network of certain process.

variables in a model. Both, the detection and isolation modules utilize a Bayesian network. The detection is made using the Bayesian network representing the probabilistic relations between the variables as in the example of Fig. 1. This is called the detection network. The isolation network is explained below.

The *basic validation* consists in the estimation of the value of a variable according to the values of other related variables. A particular variable is taken as the hypothesis while the related variables act as the evidence. Thus, the propagation in the Bayesian network provides the posterior probability distribution of the variable's estimated value, given other variables. This distribution can then be used to infer if the variable has a proper value or if it shows an abnormal behavior.

For example, for the validation of variable m in Fig. 1, a posterior probability distribution $P(m \mid t, p)$ is calculated[2]. The real value of m is compared with the probability distribution and a conclusion is obtained. However, if the validation of a variable is accomplished utilizing a faulty variable then a false conclusion is expected. A basic validation algorithm then, can only tell if a variable has a potential fault, but (without considering other validations) it can not tell if the fault is real or apparent. The *fault isolation* module distinguishes between apparent and real faults, isolating the faulty variable.

The isolation module functions as follows. When a faulty variable exists, the fault will be manifested in all the related variables. The most closely related variables for each variable in a Bayesian network are those in its *Markov blanket*. A Markov blanket (MB) is defined as the set of variables that makes a variable independent from the others. In a Bayesian network, the following three sets of neighbors are sufficient for forming a MB of a node: the set of direct predecessors, direct successors, and the direct predecessors of the successors (i.e. parents, children, and spouses) [4]. For example, consider the Bayesian model of Fig. 1. The MB of t consists of the set $\{g, a, p, m\}$, while MB of p consists only of $\{m, t\}$.

The set of variables that constitutes the MB of a variable can be seen as a protection of this variable against changes of variables outside the blanket.

[2] Notice that the value of a and g are not required since they are conditionally independent of m given t.

Additionally, the *extended Markov blanket* of a variable v_i written $EMB(v_i)$, is formed by its MB plus the variable itself. Utilizing these concepts, if a fault exists in one of the variables, it will be revealed in all the variables in its EMB. On the contrary, if a fault exists outside a variable's EMB, it will not affect the estimation of that variable. It can be said then, that the EMB of a variable acts as its protection against other faults, and also protects others from its own failure. This assumes that all the variables in the MB are instantiated. The *fault isolation* module utilizes this property to update a probability of failure vector (one for each variable) and hence, to distinguish the real faulty variable [2].

The isolation utilizes a probabilistic causal model relating the real and apparent faults [4, 5]. Fig. 2 shows the causal network corresponding to the example of Fig. 1. This is called the isolation network.

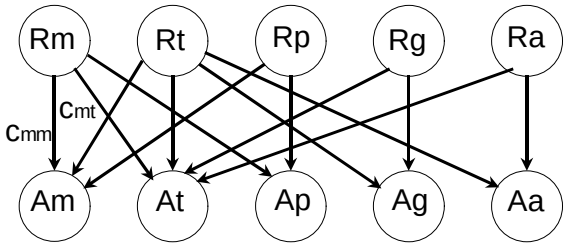

Fig. 2. Probabilistic causal model for fault isolation. R_i represents the real fault in sensor i while A_j represents the apparent fault in sensor j.

The node R_m represents the real fault in variable m while A_p represents the apparent fault in variable p. The arcs denote causality, i.e., an apparent failure in variable m (node Am) is *caused* by the existence of a real fault in one of the nodes m, t or p[3]. This set happens to be the $EMB(m)$. Thus, given more evidence about the state of apparent faults in the variables, the posterior probability value of real fault nodes will tend either to zero or one.

The probabilistic diagnosis theory developed in [2, 3] can be summarized as follows. After a cycle of basic validation of all variables is completed, a set S of apparent faulty variables is obtained. Thus, based on the comparison between S and the EMB of all variables, the following situations arise:

1. If $S = \phi$ there are no faults.
2. If S is equal to the EMB of a variable X, and there is no other EMB which is a subset of S, then there is a *single real fault* in X.
3. If S is equal to the EMB of a variable X, and there are one or more EMBs which are subsets of S, then there is a real fault in X, and possibly, real faults in the variables whose EMBs are subsets of S. In this case, there are possibly *multiple undistinguishable* real faults.

[3] The parameter c_{ij} correspond to the *noisy–or* condition [4, 3].

4. If S is equal to the union of several EMBs and the combination is unique, then there are *multiple distinguishable* real faults in all the variables whose EMB are in S.
5. If none of the above cases is satisfied, then there are multiple faults but they can not be distinguished. All the variables whose EMBs are subsets of S could have a real fault.

For example, consider the network shown in Fig. 1 and its corresponding probabilistic causal model shown in Fig. 2. Suppose a failure in the value of variable p. The basic validation of m will produce an apparent fault given that its estimation was made using an erroneous value. This imply the instantiation of node Am in Fig. 2 as true. At this point, the probability of failure will show an estimation of the most suspicious real faulty nodes. Following the procedure for all nodes, will result in the detection of apparent faults in nodes m, p, and t. Thus, instantiating those apparent fault nodes in Fig. 2 will provide the posterior probability of all real fault nodes $R_{variable}$. In this example, a high probability value in node Rp will be obtained.

This theory assumes that all the variables in the MB are insaniated when a variable is validated. This is not always feasible in model–based diagnosis. However, the influence of an instantiation of a variable outside its MB is not too strong. So it is considered that a faulty variable only affects the variables in its MB and the previous theory can be applied.

3 The diagnosis architecture

At the highest level, the architecture is a typical intelligent system, i.e., it contains a knowledge base, an inference mechanism and the output. The knowledge base is a Bayesian network representing the probabilistic model that relates all the variables. The network is constructed from a data set and using an automatic learning program. The inference mechanism is the propagation methods of elvira, and the output is a vector with the probability of failure of all the variables. From a functional point of view, the diagnosis has two tasks: fault detection and fault isolation.

Figure 3 shows the most general architecture. The learning module utilizes a data set and produces a Bayesian network to be utilized by the detection module. This module is the inference mechanism mentioned above. The detection module reads data from the plant through the real time data acquisition module. Real time data is utilized as evidence in the propagation, so apparent faults are detected as explained in section 2. The list of apparent faults is the input to the isolation module. The isolation module utilizes a causal model and produces the final output of the diagnosis system, namely, a vector with the probability of failure of every variable in the system. This vector is transmitted to the graphical user interface (GUI) for indication to the operator of the plant. Notice that the diagnosis is carried out in a cycle between the detection and isolation modules until all the variables have been diagnosed. The proposed architecture described

in this paper includes only the modules inside the dashed square. They are described in the following sections.

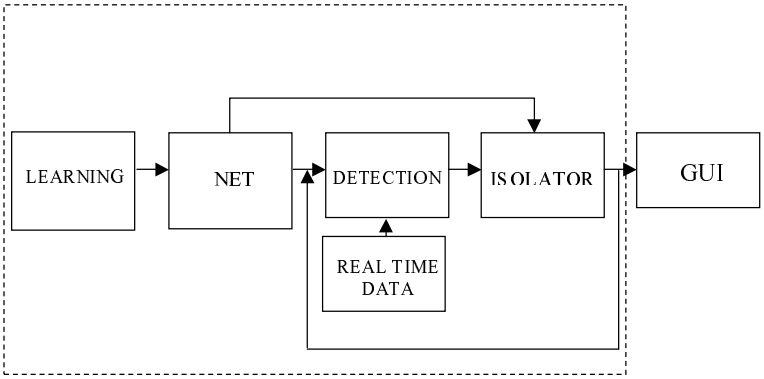

Fig. 3. High level architecture of the diagnosis system.

3.1 Learning module

The learning process starts with the acquisition of historical data from the process without faults. Normally, the variables monitored in industrial processes are continuous, i.e., discretization is required. When the sampled information has been discretized, the **K2Learning** class from elvira is utilized. There are several different learning algorithms in elvira depending the characteristics of the data. The elvira's learning module produces a file with the resulting Bayesian network in elvira's format (*detection.elv*). Figure 4 shows the steps needed in the learning process.

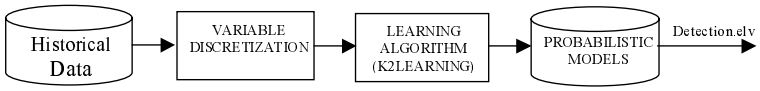

Fig. 4. Learning the probabilistic model for diagnosis.

3.2 Creating the Bayesian network

Creating a Bayesian network object in elvira requires a network description in elvira's format. This format is an ascii description of the nodes and arcs, their characteristics and the parametric information. For the detection module, the

network is obtained with the learning module mentioned above. The isolation network is formed with the detection network and the Markov blanket property of Bayesian networks as explained in section 2. Figure 5 shows that, once the elvira format has been defined, a syntactic and semantic validation is required. The syntactic analysis is made with the **parser** class, indicating if an error is found. Next, the **network** class reads the network description in memory and generates a Bnet object. The Bnet object is the basis for the probabilistic inference described next.

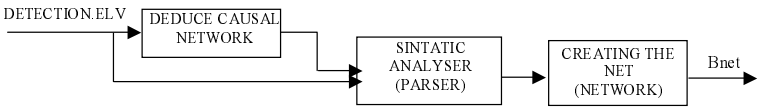

Fig. 5. Acquisition of the Bayesian networks for diagnosis.

3.3 Detection module

The detection module compiles a Bnet object it with the **bnet** class and loads the network in memory. The *in line* cycle consists in the validation, one by one, of all the variables of the model. The validation consists in the prediction of a variable's value given all the related variables. Thus, the value of the related variables are read and compared with the ranks of values generated in the discretization step of the variable. This discrete value forms the evidence for the **evidence** class. Next, the **propagation** class is called to obtain the posterior probability of the validated variable given all the evidence. The real value of the variable is compared with the probability distribution and if the deviation is higher than a specific threshold, then an apparent failure is detected. This process is repeated with all the evidence for each variable and for all variables. Figure 6 shows this process

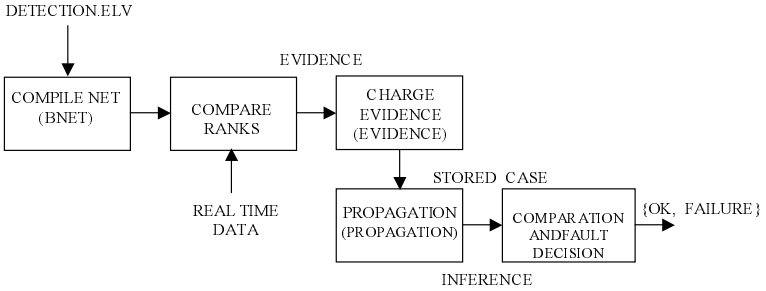

Fig. 6. Fault detection module.

3.4 Isolation module

The isolation module also utilizes a Bnet object, an evidence information and generates a posterior probability distribution. The Bnet object is obtained from the isolation causal network deduced as indicated in Fig. 5. The evidence is the decision of fault or no fault obtained in the detection module. The propagation of probabilities in the isolation module, provides the final output of the diagnosis system, namely, the probability of being faulty of all the variables in the system. The next section presents an experiment of a diagnosis of gas turbines utilizing the proposed architecture.

4 Diagnosis of gas turbines

A gas turbine consists fundamentally of four main parts: the compressor, the combustion chamber, the turbine itself and the generator. The compressor feeds air to the combustion chamber, where the gas is also fed. Here, the combustion produces high pressure gases at high temperature. The expansion of these gases in the turbine produces the turbine rotation with a torque that is transmitted to the generator in order to produce electric power output. The air is regulated by means of the *inlet guide vanes* (IGV) of the compressor, and a control valve does the same for the gas fuel in the combustion chamber. The control valve is commanded by the control system or by the operator in manual operation mode, and its aperture can be read by a position sensor. The temperature at the blade path, which is the most critical variable, is taken along the circumference of the turbine. Other important variables, measured directly through sensors are the mega watts generated and the turbine speed in revolutions per minute.

The architecture has been utilized in diagnosis experiments in a gas turbine simulator at the laboratory. Due to hardware limitations of the simulator, only 10 analog signals can be sampled every half second. The simulation executed for this experiment includes the load increasing from 2 MW to 23 MW. This procedure took about 20 minutes, so a number of 2111 records were obtained. Next, from a visual revision of the data table, three variables were discarded since they never changed during the experiment. The number of different discrete values varies depending on the precision required for each variable.

The learning module of *elvira* with the K2 algorithm [1] were executed utilizing the data table with the seven variables. Figure7 shows the resulting Bayesian network involving only six variables. The seventh variable was reported as completely independent from the rest of the variables.

The *speed* variable represents the measure of the velocity of the turbine in revolutions per second (RPS). The *power* variable is the measure of the mega watts generated, *temp* is the exhaust gas temperature in the turbine, variable *gas_press* is the gas fuel pressure, *gas_valv* is the position of gas control valve, and variable *gas_demd* is the gas valve demand.

According to the experts, the model can be interpreted as follows. The velocity or mechanical work produces mega watts and this generation produces

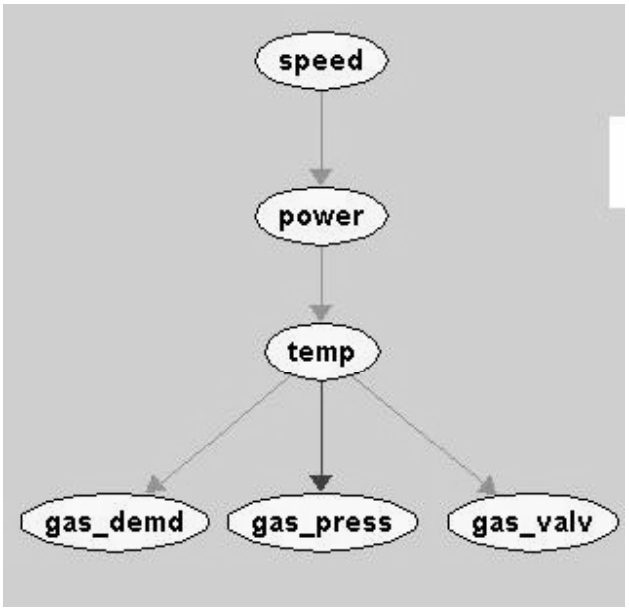

Fig. 7. Bayesian network learned from the experiment.

heat. The temperature is related with the demand of gas, the aperture of the valve and the pressure of the fuel. Notice that the direction of the arcs represents probabilistic relationship between any pair of nodes and not necessarily a *causal* relation. The variables were discretized in 3 to ten intervals according to the variation of their values along the experiment. Several tests were made with different discretization values in order to see the best result in the learning process.

Figure 8 shows the isolation network that corresponds to the probabilistic network of Fig. 7.

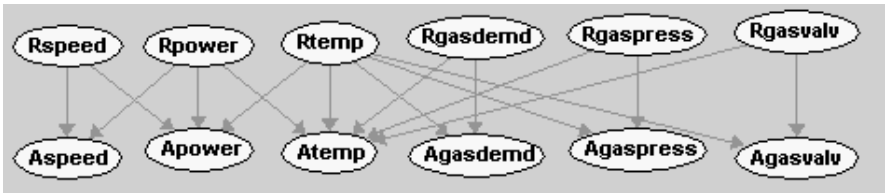

Fig. 8. Isolation network for the example of the gas turbine.

Consider the following scenario. The plant is generating and incrementing the demanded load. The demand of gas (*gas_demd*) is at 57.11 units, corresponding to state s3, the gas pressure (*gas_press*) is at 152.26, and the gas fuel position

(gas_valv) is at 56.85 units or s3 state. The exhaust gas temperature ($temp$) is at its 337.46 degrees and the gas turbine speed ($speed$) is at its 5085 rpm or the s5 state. Thus, instantiating those nodes in the network of Fig. 7, and propagating probabilities, the following results are obtained in the power variable: $P(power = s2) = 100$ %, and 0 in the rest of the power value intervals. In this case, the propagation enforces the value of $power$ variable to the interval corresponding to the state $s2$. In other cases, the posterior probability distribution presents different shapes, e.g., sharper forms.

Now, consider a failure in the electric generator or a failure in the power generation sensor, in the above scenario. Table 1 shows the diagnosis process. The second column indicates the result of the basic validation step, i.e., it provides the probability of finding an apparent fault. The third column indicates the result of the detection module according to the value of second column. This result is the value of ok or $failure$ indicated in Fig. 6. The following 6 columns indicate the posterior probability of the *Real* fault nodes in the isolation network (Fig. 8). They are in fact, the ultimate output of the diagnosis system: the probability of failure of each variable. First, variables gas_demd, gas_press and gas_valv are validated. The propagation of probabilities indicates a flat distribution in these variables, and the real value coincide in an interval with a real probability to be correct, i.e., $P(gas\_demd = correct) = 88$ %, $P(gas\_press = correct) = 54$ %, and $P(gas\_valv = correct) = 75$ %,. This implies the instantiation of nodes $Agasdemd$, $Agaspress$ and $Agasvalv$ with no apparent faults in the isolation network of Fig. 8.

Table 1. Diagnosis experiment simulating a failure in the power generation.

variable validated	$P(app.failure)$ (%)	state assigned	R_{speed}	R_{power}	R_{temp}	$R_{gasdemd}$	$R_{gaspress}$	$R_{gasvalve}$
gas_demd	88	ok	50	50	9	9	50	50
gas_press	54	ok	50	50	1	9	9	50
gas_valv	75	ok	50	50	0	9	9	9
$temp$	86	nok	50	80	0	15	15	15
$power$	20	nok	57	90	0	13	13	13
$speed$	1	ok	26	81	1	15	15	15

Notice the first row in the table. With the propagation after $Agasdemd$ was instantiated as ok, the probability of failure vector shows that the variables $temp$ and gas_demd have a 9 % probability of failure, i.e., they are free of faults while the rest of the variables are completely uncertain (50 %). After the three first variables have been validated, the results indicate that the temperature reading is securely correct, the gas variables are very low suspicious, the speed is still uncertain and the power starts to be the faulty variable. As expected from the

theory, the last step in the validation cycle confirms that the faulty variable is indeed the *power*.

5 Conclusions and future work

This paper has presented a proposed architecture for the diagnosis of industrial processes. The architecture is formed by a detection module and a isolation module. Both modules utilize probabilistic reasoning. The first, requires a probabilistic model that represents the conditional dependence and independence of all the variables in the process. The isolation module utilizes a special property of the Bayesian networks, namely the Markov blanket. A causal network is formed utilizing the first model and the Markov blanket of every variable.

The architecture was implemented utilizing the elvira software. It is a software package devoted to the probabilistic reasoning and decision support. The main classes of elvira were integrated to the diagnosis architecture and were initially tested with the diagnosis of six variables model in a gas turbine of a power plant.

The next step in this project is the fully utilization of this architecture in the diagnosis and optimization of a complete power plant. First, the selection of the most common failures reported in power plants is required. Next, the specification of all the variables involved and the probabilistic model is obtained with the learning tools Finally, with the installation of the architecture and the acquisition of the process data, faults can be detected and isolated *inline*.

References

1. G.F. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–348, 1992.
2. P.H. Ibargüengoytia, L.E. Sucar, and S. Vadera. A probabilistic model for sensor validation. In E. Horvitz and F. Jensen, editors, *Proc. Twelfth Conference on Uncertainty in Artificial Intelligence UAI-96*, pages 332–339, Portland, Oregon, U.S.A., 1996.
3. P.H. Ibargüengoytia, L.E. Sucar, and S. Vadera. Any time probabilistic reasoning for sensor validation. In G.F. Cooper and S. Moral, editors, *Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence UAI-98*, pages 266–273, Madison, Wisconsin, U.S.A., 1998.
4. J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, Palo Alto, Calif., U.S.A., 1988.
5. Y. Peng and J.A. Reggia. A probabilistic causal model for diagnostic problem solving-parts i and ii. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-17(2,3):395–406,146–162, March/April, May/June 1987.