A LAYERED, ANY TIME APPROACH TO SENSOR VALIDATION

Pablo H. Ibargüengoytia[1], Sunil Vadera[1] and L. Enrique Sucar[2]

[1] University of Salford
Department of Mathematics and Computer Science
Salford, M5 4WT,
{P.Ibar / S.Vadera}@cms.salford.ac.uk
[2] ITESM - Campus Morelos
AP C-99, Cuernavaca, Mor., 62020, Mexico
esucar@campus.mor.itesm.mx

Abstract. Sensors are the most usual source of information in many automatic systems such as automatic control, diagnosis, monitoring, etc. These computerised systems utilise different models of the process being served which usually, assume the value of the variables as a correct reading from the sensors. Unfortunately, sensors are prone to failures. This article proposes a layered approach to the use of sensor information where the lowest layer validates sensors and provides the information to the higher layers that model the process. The proposed mechanism utilises *belief networks* as the framework for failure detection, and uses a property based on the Markov blanket to isolate the faulty sensors from the apparently faulty sensors. Additionally, an any time version of the sensor validation algorithm is presented and the approach is tested on the validation of temperature sensors in a gas turbine of a power plant.

Keywords: *Uncertainty, Belief networks, sensor validation.*

1 INTRODUCTION

Current applications of artificial intelligence (AI) in real domains include different functions like automation and process control, diagnosis, monitoring etc. However, these applications require an overall process model where usually, its inputs are mainly sensors. In general, all possible functions of a computerised system require the use of reliable information in order to take the right decisions. This article proposes a mechanism for intelligent, real time sensor validation which can be utilized as a separate module that works together with other functions in industrial plants. In other words, it is assumed that a layered scheme is used in which the lowest level concentrates on validating the signals transmitted by sensors as presented in Fig. 1 [12]. Faults are detected in a decentralised and hierarchical approach, so they can be easily isolated and repaired. Additionally, suppose that the higher layers of the system represent other important and critical functions, e.g., the fault diagnosis of a nuclear plant. The intermediate

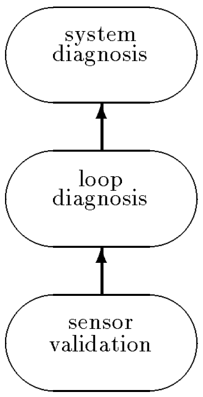

Fig. 1. Layered diagnosis architecture.

layer (loop diagnosis) may be using *model-based reasoning* to diagnose a control loop in the plant, whereas the system diagnosis layer may be utilizing a different approach. Irrespective of the approach used by the diagnosis layer, it needs to assume that the information it utilizes is accurate. This is the goal of the proposed mechanism: to provide reliable information from sensors to higher layers of a system.

This article is organized as follows. Section 2 describes the proposed sensor model and also, summarizes related work on sensor validation. Section 3 describes the architecture of the proposed model and describes the probabilistic reasoning model for fault detection. Section 4 presents an any time extension of the validation algorithm more suitable for a real time environment. Section 5 presents the results of applying the approach to a gas turbine of the Gomez Palacio power plant in México. Finally, section 6 gives the conclusions and future work.

2 THE SENSOR AND PROCESS MODELS

The input of a sensor is the value V_s which is considered unknown and inaccessible, and the output is the measurement V_m (Fig. 2). A sensor is declared *faulty* if the output measurement V_m gives an incorrect representation of the V_s [12]. A detection of a fault is made when the output of the sensor V_m exceeds some threshold, or a non permitted deviation from a characteristic trend. But, what exactly is a characteristic trend?

This question is being answered differently by many investigators. However, in all the approaches, the central idea is to *estimate* the value that a sensor must deliver based on its environment. Some examples of these environments are the following:

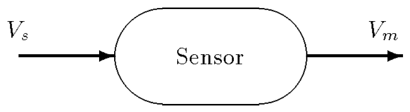

Fig. 2. Basic model of a sensor.

- history of a single signal in time,
- history of the state of the process in time,
- state of all related signals at a specific time

This estimation process is what makes the various validation approaches different. Specifically, this project uses probabilistic methods for estimation based on all the related signals at specific time instants.

As an aid to relating these different approaches, the rest of the paper will use the following terminology:

variable state (V_s): this refers to the measurand, i.e., the input to the sensor. This is the physical parameter being measured.

variable measure (V_m): this refers to the measurement, i.e., the output of the sensor. This is a numerical representation of the physical parameter state.

variable estimated (V_e): this refers to the estimated value of the variable.

sensor state (S): this refers to the condition of the sensor. This parameter has one of the binary values {correct, faulty}.

Based on these definitions, Fig. 3 shows some simplified models that can be used to represent sensor information in a physical process. These models are dependency models indicating *causality*. In (a), either the variable state *causes* the variable measure,or V_m *depends* on the value of V_s. This is the most obvious and basic model of a single sensor. Figure 3(b) shows a model including three nodes: the measure V_m depends on the variable state V_s and on the sensor state S, i.e., V_m displays a realistic representation of the variable state if the sensor is working properly ($S = correct$). Finally, since V_s is unknown and inaccessible, it is replaced with its estimation V_e in Fig. 3(c). Here, the inference on the sensor state S is dependant on the measure and the estimation. In fact, Fig. 3(c) represents the goal of this project, namely to obtain the state of the sensor based on the reading and the estimated value. In other words, this model makes explicit the conditional probability of a fault in a sensor, given the measure and the estimation, i.e., $P(S \mid V_m, V_e, \delta)$, where δ represents previous knowledge about the sensor. For example, δ might represent the mean time between failures reported by the manufacturer, the physical location of the sensor in the plant, time between the last maintenance, etc.

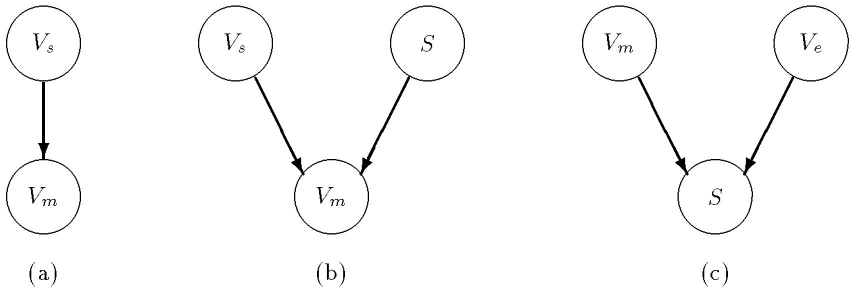

Fig. 3. Basic model of a sensor performance. (a) represents that V_m depends on V_s. (b) represents an enhanced model where V_m depends on V_s and the state of the sensor S. (c) displays the proposed basic approach, namely, S can be inferred with the values of the measure and the estimated real value.

2.1 Related work on sensor validation

Since digital computers have been accepted in industrial and specially in critical applications, the validation of sensors has been a concern for manufacturers and users. Several approaches have been proposed to detect inconsistencies in measures. The literature reports various survey papers from different application domains (e.g. [1], [12]). Recent work on this area includes the SEVA project [7] and the TIGER project [9]. The SEVA project concentrates on the design of self validating sensors using emerging techniques from digital communications for field devices. Based on microprocessors, sensors are able to detect their own failures and report the uncertainty or quality measure of their own readings. The TIGER project utilizes model-based reasoning, i.e., the system possesses a mathematical model of the process and runs a simulator in order to compare the observed output with the one estimated by the simulator. Its main function is the monitoring of the complete gas turbine, including all its parts and processes.

This article concentrates only on sensor validation utilizing probabilistic methods. Approaches proposed by Dean [3] and Horvitz [6] also have a similar focus and are summarized below.

The approach proposed by Dean presents a probabilistic model for estimating the current value of a sensor based on previous states of all sensors in the system. This approach utilises knowledge about sensor errors in terms of conditional probabilities. If $\boldsymbol{x}$ represents the system state vector, and $\boldsymbol{z}$ represents the measurement vector, then Dean represents knowledge about the performance of the sensors that produced $\boldsymbol{z}$ as a conditional probability density function, $\boldsymbol{p(x(k) \mid z(1), \ldots, z(k))}$, indicating the probability that $\boldsymbol{x}$ is the true state of the system given that $\boldsymbol{z}$ has been observed at time $\boldsymbol{k}$. This, represented graphically at a certain instant of time in the scalar case, corresponds to the simplified model of Fig. 3(a). Dean assumed that the estimated value can be determined by the *mean* of the conditional probability density function which has *white Gaussian* noise. However, this assumption is made in order to make the computation tractable even if it may not hold in the real environment.

Horvitz and co-workers described the utilisation of Belief networks for the diagnosis of gas turbines for an auxiliary power unit of a commercial aircraft. They aimed to model the whole process by using a belief network that includes sensor validation as well as the fault diagnosis process. Further, the sensor validation parts of the belief network utilizes a mixture of the different validation models outlined in Fig. 3. For example, in a fragment of their model, they represent three instances where sensors play a role. Figure 4 presents these cases. In (a), they utilise directly the value V_m as the unique source of information. In (b), they use a simplified model where V_s and S cause V_m (i.e., the model in Fig. 3(b)). Finally, in Fig. 4(c) they use a sensor state S that must obtain its value ({correct, false}) from another source (human or computerised).

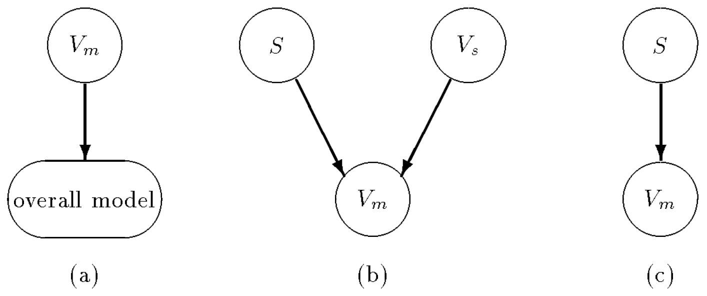

Fig. 4. Three different uses of sensor related nodes in a overall process model. In (a), the measure is utilized by the model assuming it is correct. (b) V_m depends on V_s and the state of the sensor S. In (c), the measure provided by the sensor depends only on its state.

Such an approach can be appropriate when it is not necessary to validate all the sensors with the same reliability. However, the inclusion of different sensor validation models within the full diagnosis model can result in a larger and more complex network. Hence, this article proposes a layered approach that first concentrates on sensor validation and then, on the fault diagnosis process.

2.2 Proposed Approach

Consider a system whose inputs are sensors. Utilising a simplified model of a sensor shown in Fig. 3(c), this paper proposes a general model which *separates* the sensor validation mechanism and the process diagnosis mechanism (or controlling, or monitoring etc.) as shown in Fig. 5.

The signals are validated in the first module, utilizing a sensor validation model proposed in section 3. It transmits the same signal V_m but produces a signal S which indicates the degree of belief in the measurements to the process

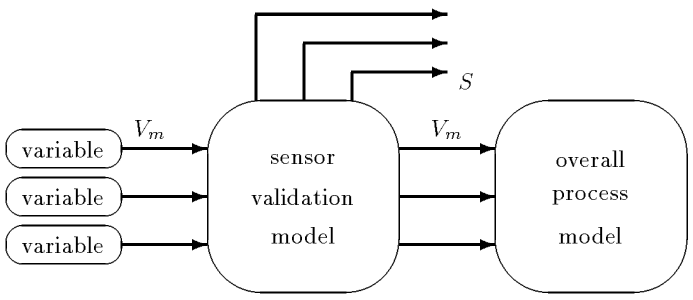

Fig. 5. General proposed model for sensor validation.

model layer [see Fig. 3(c)]. Then, this overall model may decide how and whether to utilize the sensor reading. Also, the overall model may decide if the failure is in the sensor itself, or there exists a problem in the process. Then, this corresponds to a higher level of reasoning. The next section explains the sensor validation model.

3 PROBABILISTIC SENSOR VALIDATION

The probabilistic sensor validation model utilizes belief networks. Belief networks are directed acyclic graphs (DAG) whose structure corresponds to the dependency relations of the set of variables represented in the model [10]. The nodes in this application represent the measures while the arcs represent the conditional probabilities between nodes. The structure of the network makes explicit the dependence and independence relations between the variables. The nodes generally represent discrete variables although recent work on belief networks proposes continuous representations of variables when the application requires it [5]. Alternatively, continuous variables can be discretized into some intervals according to the precision required and depending on the computational cost that is acceptable. In the current implementation, discretization is done by simply dividing the range into 10 intervals although more sophisticated approaches are available [4].

The validation algorithm considers every sensor as suspicious and obtains, one by one, the estimated value V_e of each sensor using probability propagation based on its most related variables. In belief networks, the most related variables consist of a *Markov blanket* of a node. A Markov blanket is defined as the set of variables that makes a variable independent from the others. For example, Fig. 6 shows a reduced, simplified model of a gas turbine where m represents the readings of the megawatts generated, t represents the temperature, p the pressure, g and a represent the gas and air supplied for the combustion respectively. Then, the Markov blanket of m is formed by its children t and p, and the Markov blanket of t is formed by its parent m and its children g and a. Utilizing these concepts, if a fault exists in one of the sensors, it will be revealed in all the sensors in its Markov blanket. On the contrary, if a fault exists outside a sensor's Markov blanket, it will not affect the estimation V_e of that sensor. Then, the Markov blanket of a sensor acts as its protection against others faults, and also protects others from its own failure [8].

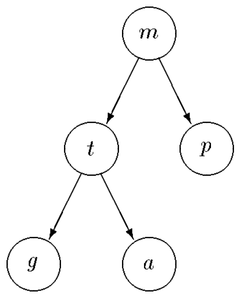

Fig. 6. A reduced Bayesian network of a turbine.

Figure 7 describes the complete process for discriminating faulty and correct sensors. The *sensors* block represents a list with all sensors to be validated. These are processed one by one by the *validation* module utilizing the following algorithm:

1. Read the value of the variable provided by the sensor.
2. Read the value of all variables that appear in the Markov blanket of the selected variable.
3. Propagate the probabilities and obtain the posterior probability distribution for the selected variable.
4. If the probability (obtained in 3) of the value acquired in step 1 is lower than a specified value (described below), return *failure*; else return *success*

The result of the validation module consists of a posterior probability distribution as shown in Fig. 8. Then, an error is detected if the difference between the real value read and the highest probability interval's value exceeds some threshold.

This algorithm provides the two lists of the potential level shown in Fig. 7, i.e., the *apparently correct* and the *apparently faulty* sensors. Then, the probabilistic reasoning can only tell if a sensor has a *potential* (real or apparent) fault, but, without considering other sensors, it can not tell if the fault is real or apparent. Thus, the next step is to distinguish between real and apparent faults, considering that one or more sensors may fail at the same time. This is performed by the *fault isolation* module described next.

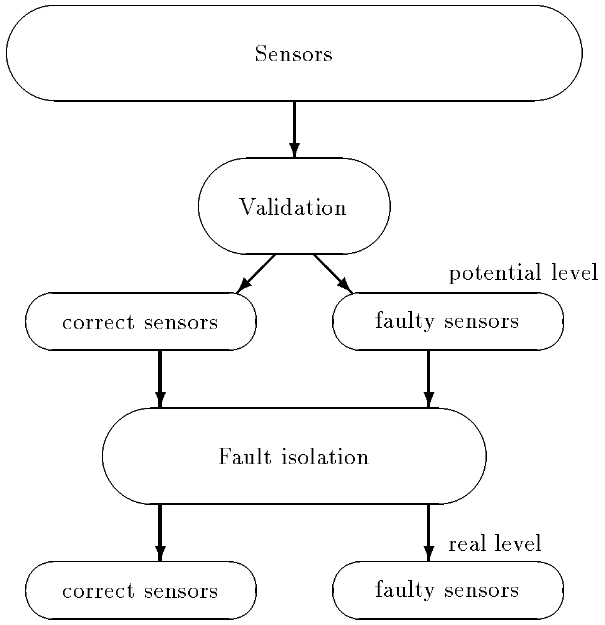

Fig. 7. Block diagram of the proposed mechanism for sensor validation.

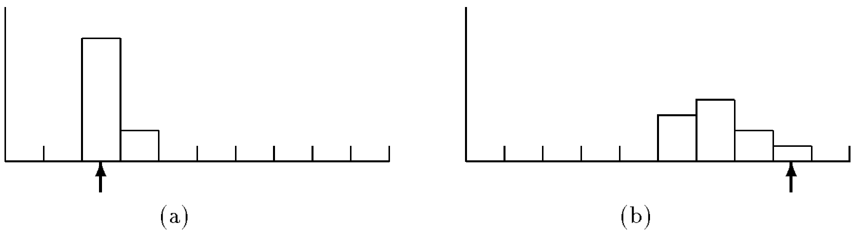

Fig. 8. Description of some possible results. The arrows indicate the interval of the real value of a sensor. (a) shows a narrow distribution where the real value coincides with the estimated. (b) shows a wider probability distribution where the real value has a low probability to be correct.

4 ANY TIME FAULT ISOLATION

The presence of a faulty sensor causes a constrained area of manifestation which forms a context. This constrained area is called the *extended Markov blanket* (EMB) of a variable. It is formed by the Markov blanket of a variable plus the variable itself. For example, in Fig. 6 the EMB of m is the set $\{m, t, p\}$, while the EMB of t is formed by $\{m, t, g, a\}$.

Suppose that there is a failure in t, and the sensors are validated following the order m, t, p, g, a. First, since t provides a wrong reading, the validation of m will report a potential fault. Then, the validation of t will also report a fault. Next, p will be validated as correct, and finally, the validation of g and a will report potential faults. At the end of the cycle, the potential fault list will contain the sensors $\{m, t, g, a\}$. This corresponds exactly with t's EMB. In other words, a sensor's Markov blanket acts like a signature that it exhibits when it is faulty [8].

This property can be used to formulate a sensor validation procedure which works in batch mode where all the sensors are examined and a faulty sensor identified. In real time applications, it is sometimes better to have some answers instead of waiting for the most accurate answer which may be too late. This problem has been addressed by a number of authors by using the concept of any time algorithms [2]. This property of an algorithm implies an incremental quality of the response with respect to time. In the sensor validation problem, the quality of the response consists of the number of sensors considered potentially correct and faulty.

An any time version of the sensor validation approach can be based on the following observations. If a sensor is found to be apparently faulty, then the Markov blanket property implies that there must be a faulty sensor in its EMB. However, if a sensor is apparently correct, there is no guarantee that all the sensors in its EMB are correct. Nevertheless, one would expect that most of the time, sensors in its EMB will also be correct. Hence, this information and the size of the Markov blanket can be used to rank the order in which the sensors are examined and leads to the following any time sensor validation algorithm.

1. Let S be a list of sensors to be validated, C be a list of validated sensors initialized to the empty list, F be a list of faulty sensors initialized to the empty list, PC and PF be lists of potentially correct and potentially faulty sensors initialized to empty lists, but where sensors that have already been tested are marked with an asterisk.
2. Select the first sensor s_i from S.
3. While S is not empty, PC and PF have sensors not yet tested do
   - apply the validation module on s_i (as given in section 3)
   - if s_i is correct,
     - then [move s_i to C, move all the sensors in its Markov blanket to PC].
     - else move all sensors in its EMB to PF.
   - if the tested sensors in PF corresponds to the EMB of a sensor, then move that sensor to F and its MB to PC (since one can expect that their failure was due to the isolated fault).
   - select the next sensor: if PF has untested sensors then select it from PF, otherwise if S is not empty then select it from S, otherwise if PC has untested sensors then select it from PC.

To illustrate the algorithm, consider the model of Fig. 6. The upper part of Table 1 shows the operation of the algorithm when there are no failures, and the lower part shows a situation when the sensor g is faulty. In the table, the final column gives a tuple that indicates the number of faulty sensors, the number of correct sensors, and the potentially faulty and correct sensors. This tuple can be viewed as a measure of the quality of the response provided by the any time algorithm at the end of each cycle. The manner in which this information is utilized will depend on the higher layer that may be application dependent. For example, a typical quality function Q may take the following form:

$$Q(F, C, PF, PC) = \alpha F + \beta C + \gamma PF + \delta PC \tag{1}$$

where $\alpha, \beta, \gamma, \delta$ are weights given to the number of sensors in each of the lists F, C, PF, PC. Suppose for presentation purposes, $(\alpha, \beta, \gamma, \delta) = (10,10,2,2)$, then Fig 9 gives a graphical representation of the assigned quality of the response against time for the examples in Table 1. This is known as the performance profile of the system.

Table 1. Example of the algorithm of the model of Fig. 6. Rows represent steps of validation whereas columns represent the content of the different lists.

validate	result	S	C	PC	F	PF	Q
		[t,m,p,g,a]	[]	[]	[]	[]	(0,0,0,0) = 0
t	OK	$[p]$	$[t]$	$[m, g, a]$	[]	[]	(0,1,0,3) = 16
p	OK	[]	$[t, p]$	$[m, g, a]$	[]	[]	(0,2,0,3) = 26
m	OK	[]	$[t, p, m]$	$[g, a]$	[]	[]	(0,3,0,2) = 34
g	OK	[]	$[t, p, m, g]$	$[a]$	[]	[]	(0,4,0,1) = 42
a	OK	[]	$[t, p, m, g, a]$	[]	[]	[]	(0,5,0,0) = 50
		[t,m,p,g,a]	[]	[]	[]	[]	(0,0,0,0) = 0
t	Fault	$[p]$	[]	[]	[]	$[m, g, a, t^*]$	(0,0,4,0) = 8
m	OK	[]	$[m]$	$[t^*, p]$	[]	$[g, a, t^*]$	(0,1,3,2) = 20
g	Fault	[]	$[m]$	$[t^*, p]$	[]	$[a, t^*, g^*]$	(0,1,3,2) = 20
a	OK	[]	$[m, a]$	$[t^*, p]$	[]	$[t^*, g^*]$	(0,2,2,2) = 28
g is	isolated	[]	$[m, a]$	$[t^*, p]$	$[g]$	[]	(1,2,0,2) = 34
p	OK	[]	$[m, a, p]$	$[t^*]$	$[g]$	[]	(1,3,0,1) = 42

5 THE CASE STUDY: A GAS TURBINE

The techniques presented in this paper are being applied to the validation of-temperature sensors in a gas turbine of a combined cycle power plant. The temperature is the most important parameter in the operation of a turbine since the optimal performance requires the operation at the maximum permitted values. However, a little increase in the temperature, over a permitted value, may cause severe damage. For this reason, the above sensor validation process is executed

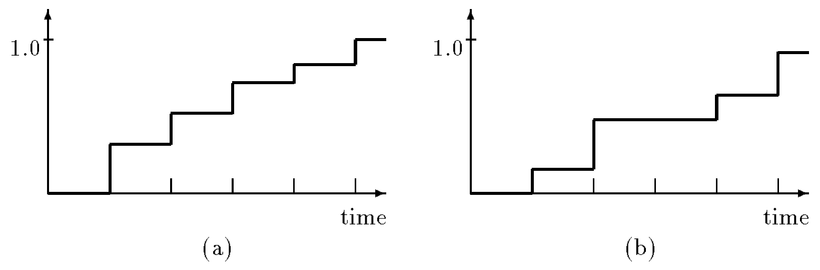

Fig. 9. Performance profile of the examples: (a) without failure, (b) with a failure simulated in sensor g.

over the set of temperature sensors across the turbine. The sensors were grouped into several measurements categories as follows:

- 6 beadings (CH1 - CH6)
- 7 disk cavities (CA1 - CA7)
- 1 cavity air cooling (AEF)
- 2 exciter air (AX1 - AX2)
- 3 blade path (EM1 - EM3)
- 2 lub oil (AL1 - AL2)

A dependency model for these temperatures was obtained utilizing an automatic learning program which, based on data, produces the optimal tree [11]. In total, the data set has 21 variables and 870 instances of the readings. These readings were taken during the start up phase of the plant. The 21 variables are continuous and were discretized for building the model, and for performing probability propagation. Figure 10 shows the probabilistic tree obtained with this data set. A tree was chosen since the inference algorithm for obtaining the posterior probabilities is faster since it depends only on the depth of the tree [10].

Notice that the dependencies can be explained as the heat *propagation* from the centre of the turbine (CH4) to the extremes. CH4 is the measure of the beading temperature which is closer to the combustion chamber, and can be modelled as the tree's root, i.e., the variable which *causes* the other variable's heating. This explanation is intuitive since, based on this data set, it is very difficult for an expert to modify it according to his experience, or to include other variables that could represent other aspects of the process.

There are two sets of experiments that are of interest in this case study. First, the overall sensor validation algorithm needs to be evaluated, and second, the performance profile of the any time algorithm needs to be determined for this application. Some initial experiments have been conducted to evaluate the validation algorithm and the results are described in the following subsection.

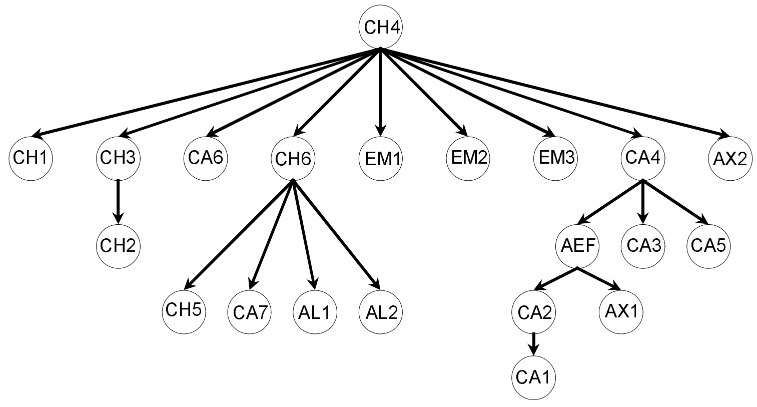

Fig. 10. Belief network of the temperature sensors in a gas turbine of a power plant.

The experiments to determine the performance profile are incomplete at the time of writing but will be available prior to the conference.

5.1 Evaluation of the sensor validation algorithm

This section presents the experimental results obtained when the algorithm is applied to the validation of the temperature sensors. The experiments were carried out in two parts: (i) to check its operation when there are no faults, and (ii) to check its operation when there is a simulated failure.

Table 2 shows the posterior probabilities of several intervals for some variables (only 6 intervals are shown, the other ones have zero probability). The interval of the variable's actual value (V_m as defined in section 2) is underlined.

Table 2. Posterior probabilities per variable per interval, with no faults.

CH4	0.00	$\underline{1.00}$	0.00	0.00	0.00	0.00
CH6	0.00	0.35	$\underline{0.37}$	0.28	0.00	0.00
CH5	0.00	0.00	0.00	0.08	$\underline{0.92}$	0.00
CA7	$\underline{1.00}$	0.00	0.00	0.00	0.00	0.00
AL1	0.00	$\underline{1.00}$	0.00	0.00	0.00	0.00
AL2	0.00	$\underline{0.88}$	0.12	0.00	0.00	0.00

The results of a simulated fault in CH6 are indicated in Table 3. Notice that all actual values are in intervals where the posterior probabilities are zero. That is, in all cases, the real value of the readings has 0% probability of being correct. For example, the line for the CH5 results shows that the real value, which corresponds to the first interval, has 0% probability. However, the posterior probability distribution shows that the correct value has a 40% probability of being in the fifth interval, and a 60% in the sixth. The results shown in the table are the sensors that constitute the CH6's EMB. The fault simulated was the reading of CH6 with its maximum value when the correct value is close to its minimum.

Table 3. Posterior probabilities per variable per interval, with a simulated fault.

CH4	0.00	0.00	0.00	0.00	0.00	0.00
CH6	1.00	0.00	0.00	0.00	0.00	0.00
CH5	0.00	0.00	0.00	0.00	0.40	0.60
CA7	0.00	0.00	0.00	0.00	0.00	1.00
AL1	0.00	0.00	0.00	0.27	0.32	0.42
AL2	0.00	0.00	0.00	0.00	0.22	0.78

These results show that: (i) in the fault free results, the real value coincides always with a highest probability interval, (ii) in the simulated fault results, the real value indicates a zero probability in and only in all the variables of the faulty sensor's EMB. However, these results are based on simulated faults on just one sensor.

6 CONCLUSIONS AND FUTURE WORK

This article has presented a layered approach to sensor validation. In comparison to other approaches, such as that of Horvitz [6], the main benefit of using a layered approach is that it enables the construction of models in a modular fashion. That is, it is easier to construct a model for sensor validation and then a model for the process than it is to construct an overall model in one step. This separation of the sensor validation layer can also result in simpler higher layer models and leave the higher layers to utilize other AI techniques.

The lowest layer, that of sensor validation, is based on the use of Bayesian networks. A Bayesian network is used to define the relationships between variables and to estimate the expected value of a sensor. The expected value is then compared with the actual reading obtained. If these measures differ then a faulty sensor is suspected. A faulty sensor is then distinguished from apparently faulty sensors by the use of a property based on the Markov blanket.

An any time version of the validation algorithm, that improves the quality of its answer incrementally, has also been presented. This any time algorithm uses the expected outcomes of a sensor together with the Markov property as a basis for ranking the order in which sensors are tested.

The approach has been implemented and is being tested on the validation of temperature sensors in a gas turbine of a combined cycle power plant. Some preliminary experimental results have been presented and suggest that the approach is promising.

In addition to further experiments, future research will attempt to use the approach on different application domains.

Acknowledgments

Special thanks to Professor F.A. Holland who provided invaluable advice in the presentation of this paper. Thanks also to the anonymous referees for their comments which improved this article. This research is supported by a grant from CONACYT and IIE under the In-House IIE/SALFORD/CONACYT doctoral programme.

References

1. M. Basseville. Detecting changes in signals and systems. *Automatica*, 24(3):309–326, 1988.
2. T. Dean and M. Boddy. An analysis of time dependent planning. In *Proc. Seventh Natl. Conf. on AI*, St. Paul, MN, U.S.A., 1988.
3. T. Dean and M.P. Wellman. *Planning and control.* Morgan Kaufmann, Palo Alto, Calif., U.S.A., 1991.
4. J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In A. Prieditis and S. Russell, editors, *Machine Learning, Proceedings of the Twelfth International Conference*, San Francisco, CA, U.S.A., 1995. Morgan Kaufmann.
5. E. Driver and D. Morrell. Implementation of continuous bayesian networks using sums of weighted gaussians. In *Proc. Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, Quebec, Canada, 1995.
6. M. Henrion, J.S. Breese, and E.J. Horvitz. Decision analysis and expert systems. *AI Magazine*, Winter:64–91, 1991.
7. M.P. Henry and D.W. Clarke. The self-validating sensor: rationale, definitions and examples. *Control Engineering Practice*, 1(4):585–610, 1993.
8. P.H. Ibargüengoytia, L.E. Sucar, and S. Vadera. A probabilistic model for sensor validation. In *Proc. Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 332–339, Portland, Oregon, U.S.A., 1996.
9. R. Milne and C. Nicol. Tiger: knowledge based gas turbine condition monitoring. *AI Communications*, 9:92–108, 1996.
10. J. Pearl. *Probabilistic reasoning in intelligent systems.* Morgan Kaufmann, Palo Alto, Calif., U.S.A., 1988.
11. L.E. Sucar, J. Pérez-Brito, and J.C. Ruiz-Suarez. Induction of dependence structures from data and its application to ozone prediction. In G.F. Forsyth and M. Ali, editors, *Procedings Eight International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE)*, pages 57–63, DSTO:Australia, 1995.
12. S.K. Yung and D.W. Clarke. Local sensor validation. *Measurement & Control*, 22(3):132–141, 1989.