15th Triennial World Congress, Barcelona, Spain

POSSIBILISTIC CAUSAL DIAGNOSIS: APPLICATION TO ENGINE DYNO TEST BENCHES

S. Boverie[****] **D. Dubois**[***] **X. Guérandel**[*]
O. de Mouzon[**] **H. Prade**[***]

[*] *D2T – SAEM, 11 rue Denis Papin, ZA Trappes Elancourt, 78190 Trappes, France*
[**] *DGEI-INSA, 135 a. de Rangueil, 31077 Toulouse Cx 4, France*
[***] *IRIT, UPS, 118 r. de Narbonne, 31062 Toulouse Cx 4, France*
[****] *Siemens VDO Automotive S.A.S., 21 avenue du Mirail, BP 1149, 31036 Toulouse Cedex, France*

Abstract: A causal diagnosis solution relying on experts' knowledge concentrates on the different malfunctions that may disturb the data acquisition process and tells the engine test bench tuning engineer which malfunction has occurred or which malfunctions are most likely suspected. The use of fuzzy sets and possibility theory provides better feedback and knowledge representation. The general architecture of the system is described, and a prototype of the fault-diagnosis part of this sytem is presented. It concerns the implementation of an (off-line) automatic knowledge formalization system and the implementation of the (on-line) possibilistic causal diagnosis process. *Copyright © 2002 IFAC*

Keywords: Diagnosis, engine dynamometer, uncertainty, fuzzy sets, possibility theory

1. INTRODUCTION

This paper presents a prototype implementation part of a long term research and development project, which aims at improving the calibration of car engine ECUs (ECU: Electronic Control Unit), on engine dyno test benches, by detecting malfunctions when they occur. This project is named *BEST*, standing for Bench Expert System Tool. Here, the implementation process of BEST only is presented. See (Boverie *et al.*, 2002) for more details on the general approach to diagnosis developed for BEST.

The engine dyno context, the needs and expected benefits (regarding diagnosis expert system BEST) are explained below (Section 2). Section 3 gives BEST's general architecture and concentrates on the fault-diagnosis part, which is implemented in a prototype. Section 4 presents the representation framework of the knowledge on malfunctions (faults) and the automatic formalization system developed in order to collect knowledge and use it very fast in the diagnosis process. Section 5 presents the approach first outlined in (Dubois *et al.*, 1999) and implemented on a prototype. Finally some perspectives are given in conclusion.

2. THE ENGINE DYNO

In one century, diesel and gasoline cars have gone from the carburettor to the electronic injection (ECU-controlled). More and more complex strategies have been implemented to answer to increasing constraints (pollutant regulations, vehicle behavior, new engine configuration: direct injection, diesel common rail, Variable Valve Timing, Electric Controlled Throttle...) and an increasing number of variables must be taken into account.

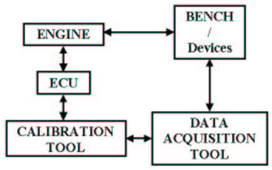


Fig. 1. The calibration process

For each new engine or new version, the control strategy parameters have to be calibrated in order to fit the requirements. This is done through a large amount of testing and tuning. These tests can be done on the engine dyno bench, on the chassis dyno or directly on vehicles. During all these processes, data acquisitions are made, and then used to define calibrations.

2.1 *The calibration process*

Figure 2.1 shows the calibration process on an engine dyno. The calibration of the ECU is performed thanks to a calibration tool. Basically, the ECU gets measurements from engine sensors (e.g., Mass Air Flow, Engine Speed . . .), computes other intermediate variables and finally tells the engine which amount of fuel should be injected and what the spark advance should be. The data used for the calibration process are recorded by the calibration tool. They are provided by engine sensors and ECU but also by the engine dyno sensors, as well as additionnal devices. That is the *System*. In the following, *System* stands for the engine, the engine dyno, all sensor sets and additionnal devices.

2.2 *Data acquisition*

Before doing any acquisition, the global conformity of the system has to be checked. It must be in accordance with the specification and the methodology. This global check includes physical verification (sensors, fuel consumption measurements, gas analysis. . .), but also system configuration (i.e. inhibited system functions). As sensors become more numerous and the strategies more complex, this global check requires more time and so the tuning engineer has less time for the testing itself and its methodology. The result of this situation is that test duration is increasing, and the reliability is degraded.

Then, while performing the tests, an on-line verification must be done to garantee the acquisition validity. Again the tuning engineer can check the validity of measurements manually while running. He checks single-parameter raw thresholds, coherence of some parameters with standard values, as well as coherence between several parameters. He also compares mesurements with those of previous tests. Nevertheless, today it is nearly impossible for the engineers to ensure the global coherence on-line (real time). Most of the time, when there is a problem, it is discovered during post processing data treatment. Quite often the test has to be performed again.

This paper describes the need for a(n expert) system, which takes into account all of these issues linked with global system conformity.

2.3 *Needs and expected benefits*

Today, 10 to 20% of the manual tests must be reworked due to bad acquisitions, bad software configuration. . . Most of the time malfunctions are due to dyno bench environmental problems (gaz analysis, fuel balance. . .). So wrong acquisitions should be detected right away and the malfunction that occured should be identified as soon as possible in order to correct it quickly and make sound acquisitions again.

Some common and simple malfunctions are already easily identified by engineers. Yet others require time-consumming searches for their origin and symptoms whenever they occur. Indeed, engineers cannot keep in mind all the information and past experiences. Moreover they cannot watch in real time all the (numerous) measured channels.

In order to cope with such checkings, an expert system, BEST, has been considered and a prototype has been developed. BEST has to perform global coherence checking, in the same way as for manual tests. It should be able to *detect* and *identify* malfunctions *as soon as* they appear: That is *on-line detection*.

3. GENERAL STRUCTURE

BEST represents a huge amount of work and investment. It gathers several functionnalities divided into different modules for step by step development and validation. This Section presents the general architecture of BEST and the part for which a prototype has been developed. Some of the ideas underlying this architecture can be found in other approaches to industrial diagnosis (e.g., (Cordier *et al.*, 2000)).

3.1 *Project's architecture*

Figure 3.1 presents the different modules:
- **FORM**, which enables the experts to formalize their knowledge, using fuzzy rules.
- **ESO**, which Extracts, Sorts and Organizes the rules w.r.t bench/engine environment specificities.

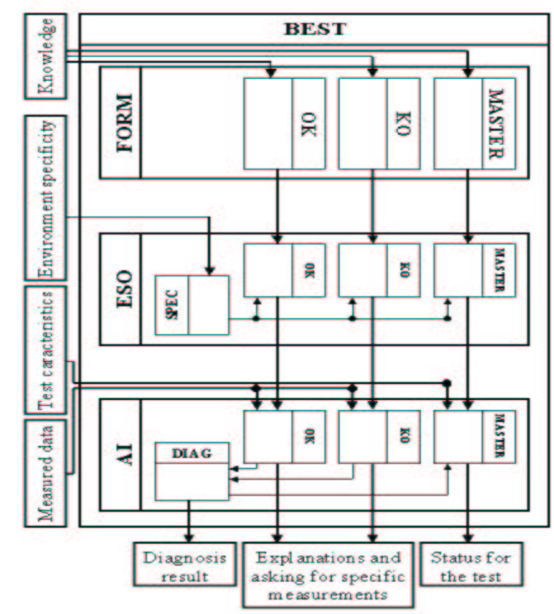


Fig. 2. The architecture of BEST

• **AI**, which is the Artificial Intelligence part performing the diagnosis by using the extracted rules and the measurements made on the engine.

Each module, contains the 3 following submodules: OK (diagnosis based on models of normal behaviour), KO (diagnosis based on models of malfunctions) and MASTER (the supervision rules). In the AI module, OK and KO diagnosis are aggregated in DIAG. BEST_AI_OK tells which models of normal behaviour are not reached and BEST_AI_KO tells which malfunctions have been identified. The use of two different diagnosis is safer in regard to incompleteness and possible inconsistency of some models. Besides, both diagnosis (OK and KO) can give some feedback explaining why a model of normal behaviour was not reached and why a malfunction was selected. They may also ask for other specific measurements in order to improve their diagnosis. Finally, MASTER decides wheather the test may continue, should use another computation for some variables, or should stop. This decision is taken according to the supervision rules, the specificity of the test (telling which variables are necessary in this test), the two diagnosis (pointing at the source faulty variables) and a dependency graph (which gives the induced faulty variables).

3.2 *Implementation*

A prototype has been developed for the whole KO part of the BEST project. The main expectation is to have a demonstrative application which can detect and identify malfunctions as soon as they appear on engine dyno test benches. Recording and formalizing the available knowledge is a key step of this project and a preliminary standard form has been defined to describe the malfunctions (Boverie *et al.*, 2001).

The entire prototype was developed under Windows NT in Visual C++ with MFC library for the user interface model. It implements the document/view architecture using MDI (Multiple Document Interface) template: the application can have two or more documents open for editing at once. Moreover, the database was designed under Access and was used through the DAO (Data Access Objects) application programming interface.

The solution implemented is based on the generation from a knowledge database of a fuzzy rule file consistent with test specificities and environment. This file is then used by another tool to perform the diagnosis on engine dyno computers. So, the prototype is made of two standalone applications: one for off-line formalization and selection of knowledge (Section 4) and the other for on-line diagnosis (Section 5).

4. OFF-LINE TOOL

The main interface is made of two parts:
• A *FORM part* which concerns the database management, and consists of a workspace with a search engine (Section 4.1).
• An *ESO part* which concerns the files management. By default, the workspace shows a list of all malfunctions already created in database, each one is detailed as a tree of symptoms (Section 4.2).

4.1 *FORM part*

First of all, it is a tool for the enlargement of the knowledge database to obtain more consistent rules. The user has to populate database with engine experts' knowledge about the malfunctions and their effects. It is generally a laborious and repetitive work. For each malfunction identified, the following three tasks have to be done:

• Give the malfunction definition: the user has to enter a unique definition with four characteristics: a group, a type, an identification and the nature of default (some choices are proposed for each).

• Give the malfunction structure: according to the algorithm, the user has to describe the malfunction with symptoms built with two level connections (AND/OR logical operators with a level of confidence and weighting).

• Give the symptom details with four elements: an attribute definition (mathematical function for the observed anomaly), a possibility level (probability to observe the attribute within a defined range), some conditions (operations having an influence on the way the attribute can appear), and an environment (bench and engine specificities).

It is important to notice that a symptom is described for a specific environment. So, a malfunction can have the same symptom several times just because of different environments.

4.2 *ESO part*

The purpose of ESO is to Extract, Sort and Organize the rules depending on the environment of the engine dyno test bench. The prototype implements ESO as a functionality of the BEST_FORM_KO module.

It consists of files that are used to give knowledge to the AI core which performs the malfunctions diagnosis by itself. When a new ESO file is created, the user has to parameter its environment. Only malfunctions of the workspace having compatible symptoms with this environment are automatically added into the file (through a serialization mechanism). For the incoherent symptoms, the user can choose to modify their environment in order to make them compatible or to remove them. The generated files are very efficient but unreadable (of course, they can be easily modified with the tool).

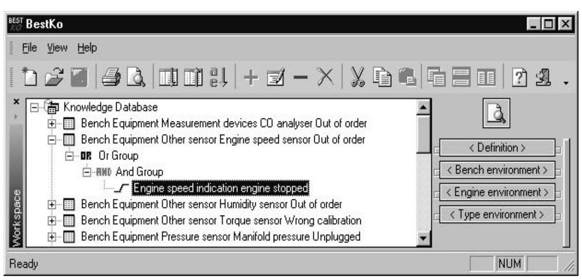
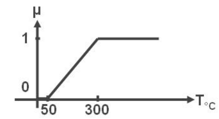
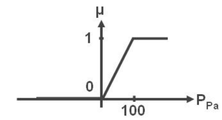
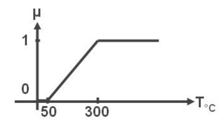
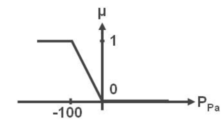
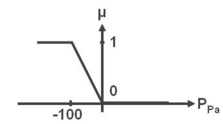


Fig. 3. Main screen

5. ON-LINE TOOL

This Section describes the basis of the methodology on which relies the diagnosis process of the prototype. It first defines some notations (Section 5.1) for the malfunctions, attributes, causal fuzzy rules (formalized knowledge) and observations (also possibly fuzzy). Then, the diagnosis is developed on the idea of consistency (Section 5.2) and finally it is refined with abduction (Section 5.3). A toy example is carried out at each step. A prototype implementation of this diagnosis process is then presented (Section 5.4).

Table 1. Fuzzy Causal Knowledge

Malfunction	Inlet Temperature	Exhaust Gas Back Pressure
m_1		
m_2		
m_3		

5.1 *Notations*

Let $\mathcal{M}$ be the set of all (known) possible malfunctions and $\mathcal{A}$ be the set of the n observable attributes: $\{X_1, \cdots, X_n\}$. Let $m \in \mathcal{M}$ and $i \in \{1, \cdots, n\}$, then π_m^i denotes the possibility distribution (Zadeh, 1978), giving the (more or less) plausible values for attribute X_i when malfunction m (alone) is present. Let U_i be the domain of X_i, so $\pi_{\mathbf{m}}^{\mathbf{i}} : U_i \longrightarrow [0,\ 1]$. $\mathcal{K}_{\mathbf{m}}^{\mathbf{i}}$ will be the fuzzy set corresponding to possibility distribution π_m^i. It represents what is known about the effects of malfunction $\underline{m}$ on attribute $X_{\underline{i}}$. $\mathcal{K}_{\mathbf{m}}^{\mathbf{i}}$ is also called *effect*, or *symptom, of m on X_i*.

For instance, let $\mathcal{M} = \{m_1, m_2, m_3\}$, $X_1 = T$ (the inlet temperature) and $X_2 = P$ (the exhaust gas back pressure). Table 1 summerizes the knowledge concerning the effects of these 3 malfunctions on those two attributes:

- m_1 gives a high inlet temperature and a positive exhaust gas back pressure.
- m_2 gives a high inlet temperature and a negative exhaust gas back pressure.
- m_3 (has no effect on inlet temperature and) gives a negative exhaust gas back pressure.

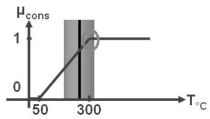
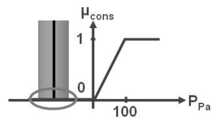
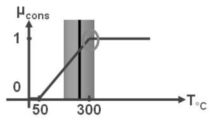
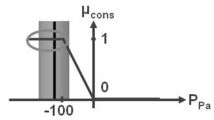
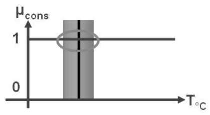
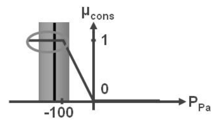
The observations may also be imprecise (or uncertain). $\mu_{\mathcal{O}_i} : U_i \longrightarrow [0,\ 1]$ is the possibility distribution, which gives the (more or less) plausible values for the observed value of attribute X_i. $\mathcal{O}_\mathbf{i}$ denotes the fuzzy set corresponding to possibility distribution $\mu_{\mathcal{O}_i}$. It expresses the imprecision (or uncertainty) of the observations (coming from sensors). In other words, it represents the possible actual values for attribute $X_{\underline{i}}$.

In the toy example, the imprecisions of the observations are represented with crisp sets (Table 2).

$\mathcal{K}_m^i$ and $\mathcal{O}_i$ both express imprecision, when they contain more than one element. Yet, they give information of two highly different types:

- imprecision for $\mathcal{O}_i$ can be “controlled”: Changing the sensors would give more precise (but may be more expensive) or less precise observations.

Table 2. Using the consistency index

Malfunction	Inlet Temperature	Exhaust Gas Back Pressure
m_1		
m_2		
m_3		

• imprecision on $\mathcal{K}_m^i$, on the contrary, cannot be reduced (or changed) that easily: It depends on the available knowledge about the *System* only.

Note that when attribute X_i is not yet observed, its value is not known, and it could be any value of U_i: $\forall u \in U_i,\ \mu_{\mathcal{O}_i}(u) = 1$. Similarly, when malfunction m has no known effect on attribute X_i, all values are allowed: $\forall u \in U_i,\ \pi_m^i(u) = 1$.

In fact, for the knowledge representation, the experts are very often more comfortable in expressing their confidence in some values they consider highly possible or, on the contrary, totaly impossible. So, for continuous attributes, the experts only need to tell what they know best (values of possibility 0 and 1) and π_m^i is then computed to follow the given information and to be continuous and piecewise linear (see Table 1). For discrete attributes, we can have different levels of possibility (e.g., from 0 to 1 by step of 0.1 units).

5.2 *A consistency-based index*

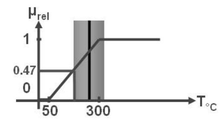
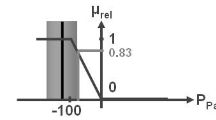
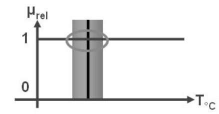
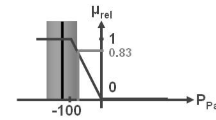
A consistency-based index has been defined: $\mu_{cons} : \mathcal{M} \longrightarrow [0,\ 1]$, which enables to discard observation-inconsistent malfunctions ($\mu_{cons}(m)$ close to 0).

The possibility distribution attached to $\mathcal{O}_i \bigcap \mathcal{K}_m^i$ is defined by: $u \longmapsto \min(\mu_{\mathcal{O}_i}(u),\ \pi_m^i(u))$ and tells how much the observations and the knowledge on malfunction m are consistent. The elements of highest possibility in this intersection give the *consistency degree between $\mathcal{O}_i$ and $\mathcal{K}_m^i$*: $\sup_{u \in U_i} \min(\mu_{\mathcal{O}_i}(u),\ \pi_m^i(u))$. The consistency degree for any malfunction m with the observations is then given according to those of $\mathcal{O}_i$ and $\mathcal{K}_m^i$ (for each attribute):

$$\mu_{\mathbf{cons}}(m) = \min_{i=1}^{n} \sup_{u \in U_i} \min(\mu_{\mathcal{O}_i}(u),\ \pi_m^i(u)).(1)$$

So the toy example leads to Table 2. Here, m_1 is discarded by μ_{cons} as the measured exhaust gas back pressure is incompatible with the presence of

Table 3. Using abduction

Malfunction	Inlet Temperature	Exhaust Gas Back Pressure
m_2		
m_3		

m_1. Yet, m_2 and m_3 are both absolutely consistent with the observations. Should one of them be a better explanation of the observed symptoms?

In case of too incomplete knowledge, μ_{cons} might not be sufficient in order to select a small enough number of malfunctions. So, a second index is required in order to refine μ_{cons} and bring a better conclusion: find, among the undiscarded malfunctions, which one is most relevant to the observations.

5.3 *An abduction-based index*

A malfunction is more relevant to the observations when its effects have been observed for sure. That is: $\mathcal{O}_i \subseteq \mathcal{K}_m^i$ (for crisp sets). In order to single out most relevant malfunctions, fuzzy inclusion of $\mathcal{O}_i$ in $\mathcal{K}_m^i$ has to be defined.

Inclusion can be defined by implication (for crisp sets, $A \subset B$ is equivalent to $\forall x, x \in A \Rightarrow x \in B$). So, several fuzzy implications ($\rightarrow$) have been checked for this purpose. Thus a second index is defined to evaluate the relevance of a malfunction:

$$\mu_{\mathbf{rel}}(m) = \min_{i=1}^{n} \inf_{u \in U} \mu_{\mathcal{O}_i}(u) \rightarrow \pi_m^i(u). \quad (2)$$

The worst implication degree tells the extent to which the malfunction is relevant to the observations. So μ_{rel} selects the most relevant malfunctions (μ_{rel} close to 1).

Dienes' strong implication ($a \rightarrow_D b \doteq \max(1-a,\ b)$) was chosen because it is the most dicriminating and it keeps the following natural crisp property: if $\mathcal{O}_i \subseteq \mathcal{K}_m^i$, then $\mathcal{O}_i \bigcap \mathcal{K}_m^i \neq \emptyset$. That is: $\mu_{cons} \geq \mu_{rel}$.

This index has abductive caracteristics as it selects m from the knowledge of the effects of m and the observation of the effects of m.

The toy example then leads to Table 3 in order to classify equally-consistent malfunctions (here m_2 and m_3). So m_3 is the best explanation, as it has the highest confidence level concerning the presence of its symptoms (0.83 vs. 0.47 for m_2).

Note that the use of μ_{rel} is linked to the fact that observations are imprecise (indeed, $\mu_{rel} = \mu_{cons}$ in case of precise observations). Yet, a totally precise observation is feasible only for discrete attributes. In case of continuous attributes (e.g., temperature, pressure), the observations given by sensors always have an imprecision (even if it can be made very small with high precision sensors).

As a conclusion, the diagnosis is first based on μ_{cons} in order to discard and rank the malfunctions and then on μ_{rel} in case of twin first malfunctions (as in the toy example). See (Boverie *et al.*, 2002) for a more complete discussion on the use of fuzzy sets in this diagnosis process and the extension to multiple-fault diagnosis (not yet implemented).

5.4 *Prototype implementation*

The first prototype developed (Boverie *et al.*, 2001) has been adapted to interface with the off-line formalisation tool through the new ESO files format (a deserialization mechanism extracts all the causal fuzzy rules into memory). It has no impact on the AI core algorithm which has only minor changes. However, some improvements have been implemented to make the application more useful. There are now two modes:

• An efficient diagnosis mode, which consists of a small icon sitting in the system tray of Windows NT taskbar. While performing on engine dyno test, the prototype becomes active and works without human intervention. When an event occurs, i.e. detection of the possible presence of a malfunction, a popup window is automatically opened and gives some information and advice on malfunctions and symptoms involved in the problem.

• A degrade debug mode, which displays a bar-graphs window and a console window where each event (attribute calculus, intermediate results, false alarms, ...) is logged and can be saved on a text file.

6. CONCLUSION

This project has lead to progress in the diagnosis based on expert knowledge. It gives a complete base for the diagnosis, from the computer-assisted recording and formalization of information on malfunctions to the detection of their presence.

The diagnosis prototype (which was first developed and has now been improved) already showed the efficiency of the fuzzy causal diagnosis methodology. Further improvements should enable the detection of multiple malfunctions and "cascading" malfunctions (Boverie *et al.*, 2002), in connection with the formalization tool.

The formalization tool, which has been more recently implemented, enables to define (single) malfunctions and symptoms, using convenient windows and a subset of the human natural language (subset of English). The diagnosis prototype can then use this knowledge, directly from the database. Yet, some necessary functionalities should be implemented in the future in order to reach an industrialisation process: administrative tasks (user's rights on data, validation procedure), network capabilities to allow concurrent access to the database, and rollback on the database.

Moreover, the OK part is under study. The goal is to develop a complete prototype of BEST, including the master supervision rules. Indeed, these rules are a key step towards a controlled automatic calibration of the ECUs, an idea which is rising in the automotive community.

REFERENCES

Boverie, S., D. Dubois, O. de Mouzon and H. Prade (2002). Possibilistic causal diagnosis: From single to multiple faults. In: *Proc. 15th IFAC World Congress*. Barcelona.

Boverie, S., D. Dubois, X. Guérandel, O. de Mouzon, C. Peyrau and H. Prade (2001). Using fuzziness for causal diagnosis in engine dyno test benches. In: *Proc. IFAC Conf. on Advanced Fuzzy/Neural Control 2001*. Valencia. In press.

Cordier, M.-O., P. Dague, M. Dumas, F. Lévy, J. Montmain, M. Staroswiecki and L. Travé-Massuyès (2000). Ai and automatic control approches of model-based diagnosis: links and underlying hypotheses. In: *Safeprocess*. Vol. 1. pp. 274–279.

Dubois, D., M. Grabisch and H. Prade (1999). A general approach to diagnosis in a fuzzy setting. In: *Proc. 8th Int. Fuzzy Syst. Assoc. World Conf. (IFSA 99)*. Taiwan. pp. 680–684.

Zadeh, L. A. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* **1**, 3–28.