FAULT DIAGNOSIS OF COMPLEX SYSTEMS BASED ON MODULAR KNOWLEDGE BASE AND INFORMATION COMPRESSION

Gancho Vachkov
Department of Reliability-based Information Systems Engineering,
Faculty of Engineering, Kagawa University
Hayashi-cho 2217-20, Takamatsu-Shi, Kagawa-ken 761-0396, Japan
E-mail: vachkov@eng.kagawa-u.ac.jp

KEYWORDS
Modular Knowledge Base, Fault Diagnosis, Information Compression, Neural-Gas Learning, Similarity Degree

ABSTRACT

A fault diagnosis method for complex dynamic processes and systems is proposed in the paper. It uses a special modular knowledge base, which is a collection of modules for some typical faulty and normal conditions of the system. Each module is considered as a kind of *compressed information model*, which keeps in a compact form the most representative characteristics of the original data, collected for the concrete system condition. Here a modified version of the neural-gas learning algorithm for creation of all compressed information models is proposed in the paper, where a preliminary assumed number of neurons is used. The collected data from the current operation of the system are also transformed into a respective compressed information model by the same learning algorithm.
The proposed fault diagnosis method is an evaluation procedure for the similarity degree between the compressed information model for the current operation and all the modules form the knowledge base. Here three different measures of similarity are used, with the "center-of-gravity distance" showing the best results.
Real experimental data taken from different operations of a hydraulic excavator are used in the paper to analyse and prove the applicability of the proposed fault diagnosis method.

INTRODUCTION

The methods for fault diagnosis and condition monitoring play important role in the safe and faultless operation of complex systems and machines, such as hydraulic excavators and other construction machines, chemical plants, heavy trucks etc.
The high dimensionality of the data obtained from a large number of sensors, as well as the large amount of the obtained data during the machine operation create real difficulties in solving the fault diagnosis problem. Here an appropriate "information compression" method could be very helpful for efficient saving and processing of the bulky data. Then, a respective fault diagnosis, method, being able to work with this "compressed information" will be needed.
The most often used approaches for fault diagnosis are the model-based approach and the pattern recognition approach with many methods and algorithms developed for each of them.
The model-based fault diagnosis approach (Gertler 1999) deals with parameters estimation and state estimation of the system. Here the model parameters of the current system are compared with those of the "healthy system" in order to make a decision about the faulty state of the system.
The pattern recognition approach for fault diagnosis (Zhao 2004) uses certain responses of the system from different operating conditions and tries to classify them into groups (clusters or rules). that represent clearly some faulty conditions. While this approach is easier to apply since it does not need a rigorous model of the system, it has the disadvantage of being data-dependent. It means that it tends to "remember" the specific data-set used for training and therefore lacks good generalization ability.
Different applications of the Fuzzy Logic and Fuzzy Models (Isermann 1998) and also the Neural Networks (Bishop 2003, Kohonen 2002) have improved the general pattern recognition approach. Here both types, namely the supervised and unsupervised learning have been successfully applied.
In this paper we are dealing with the fault diagnosis problem for dynamic processes and complex systems working under differet conditions. The specific difficulty here is that there could be different dynamic behaviors of the complex system that still represent a "normal (healthy) state" of the system. . Such typical examples are a "healthy" machine, working under many different (and even time-varying) loads. Obviously, it would be not practical to apply some fault diagnosis methods (Zhao 2004) that try to remember the process behaviors in the direct form of transient processes for many faulty states. Another (more "compressed way") of remembering the system behaviors is needed
In this paper we propose an efficient computational strategy for analysis and detection of changes and deterioration trends in operation of construction machines. It is a *knowledge-based fault diagnosis,* which uses the concept of information compression of the original large number of "raw data" into a smaller portion of "information granues" (Pedrycz 2005) in the form of neurons in the multidimensional space.

THE PROPOSED FAULT DIAGNOSIS METHOD AND SYSTEM

The general structure of the fault diagnosis method and system proposed in this paper, is shown in Fig. 1. This is a knowledge-based fault diagnosis concept, in which the preliminary created Modular Knowledge Base (MKB) plays a critical role for the correctness and plausibility of the results from the fault diagnosis. The MKB consists of a collection of L preliminary created Knowledge Modules (KM), each of them keeping the most essential features of one typical (*faulty* or *normal*) operation of the complex system (machine). This is done by a special learning procedure which creates the so called *Compressed Information Model* (CIM) by using the raw data, collected for this specific operation of the complex system. The CIM is considered as a kind of Neural Model which represents the information, contained in the raw data set in a highly compressed (usually at least *20* times) form. The method for this information compression is explained in details in the sequel of the paper.

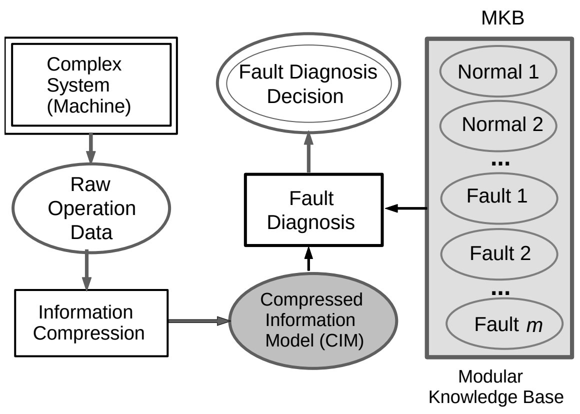


Figure 1: The Proposed System for Fault Diagnosis by Using Compressed Information Models

When a current operation status of the complex system should be diagnosed, first of all M *raw operation data:* $\mathbf{x}_s=[x_{s1},x_{s2},...,x_{sK}],\ s=1,2,...,M$ are taken as measurements from K sensors, which creates a data set with M data points in the K-dimensional space. The time period for such data collection should be long enough to ensure that sufficient "representative" information about the working condition of the machine is obtained. Depending on the machine dynamics and also on the sampling period of the sensors, usually many thousands of data are collected for further analysis and diagnosis. Then an information compression method is used to produce the CIM for this operation condition, according to Fig. 1.

The proposed fault diagnosis method can be generally considered as a computational procedure for evaluation of the *degree of similarity* between each of the knowledge modules (KM) in the knowledge bas (MKB) and the compressed information model (CIM) for the current operation. The "most similar" KM to the current CIM suggests that the current operation of the complex system most "resembles" the fault described by this KM.

The method proposed here is similar to the previously proposed fault diagnosis method (Vachkov, 2005) which compares directly the raw data set from the current operation with the compressed models from the Knowledge Base. The main difference is that here, according to Fig. 1., we evaluate the *similarity degrees* between pairs of *two* compressed information models, namely the current CIM and a KM from the MKB.

A feasible method for evaluating the similarity degree between two CIMs could be the use of the reciprocal measure of the distance (the *degree of dissimilarity*) between two CIMs, as shown graphically in Fig. 2. Here 100% denotes the *shortest distance* (the biggest similarity) between the SIM for the current operation and a KM from the MKB.

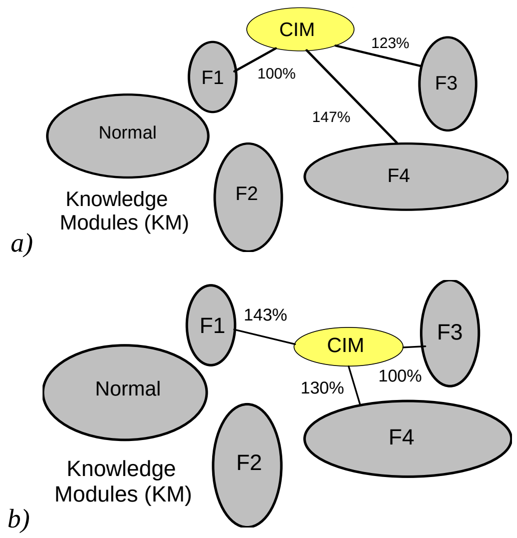


Figure 2: Illusration of the Fault Diagnosis Idea by Using the Notion of *Distance* between two CIMs.

COMPUTING THE DISTANCES BETWEEN THE COMPRESSED INFORMATION MODELS

The compressed model CIM, created from M raw data by a learning algorithm (explained in the next Section of the paper), consists of N neurons with two groups of parameters, as follows: Neuron *Centers* (*Locations*) $\mathbf{C}=\{\mathbf{c}_1,\mathbf{c}_2,...,\mathbf{c}_N\}$ in the K-dimensional space and Neuron *Weights* $\mathbf{G}=\{g_1,g_2,...,g_N\}$, where

$\mathbf{c}_i=[c_{i1},c_{i2},...,c_{iK}],\ i=1,2,...,N$ and

$$0<g_i\le 1,\ i=1,2,...,N \text{ with } \sum_{i=1}^{N} g_i=1.$$

Every CIM is considered as a *data cluster* and there could be different ways of computing the distances between any two CIMs (any two clusters), as studied in (Pedrycz, 2005). In our fault diagnosis approach, we use three methods for computing the distances between the Compressed Information Model **A**, which represents the current operation of the complex system and a given Knowledge Module **B** from the MKB. The number of neurons in **A** and **B** are denoted as N_A and N_B respectively.

a) *Center-of-Gravity Distance* (COGD) :

$$COGD(\mathbf{A},\mathbf{B}) = \sqrt{\sum_{j=1}^{K}[COG_A(j) - COG_B(j)]^2}, \quad (1)$$

with the *Center-of-Gravity* (COG) of each model computed as:

$$COG(j) = \sum_{i=1}^{N} c_{ij} g_i \Big/ \sum_{i=1}^{N} g_i, \; j=1,2,...,K. \quad (2)$$

b) The *Weighted Average of the Minimal Distances* (WAMD):

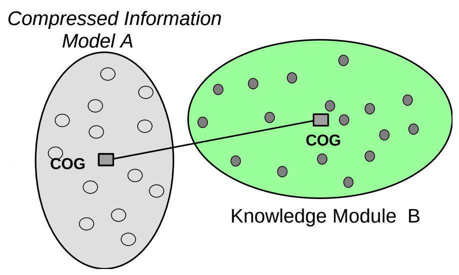
$$WAMD(\mathbf{A},\mathbf{B}) = \sum_{p=1}^{N_A} D_{p\min}\, g_p g_{q*} \Big/ \sum_{p=1}^{N_A} g_p g_{q*}, \quad (3)$$

where $D_{p\min}$ is the minimal *Euclidean* distance between the *p*-th neuron from **A** and a neuron from **B**, found with the q*-th neuron from **B.**

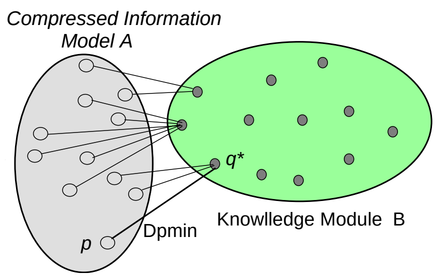
c) The *Weighted Average of All-Pair Distance*s: (WAPD):

$$WAPD(\mathbf{A},\mathbf{B}) = \sum_{p=1}^{N_A} \sum_{q=1}^{N_B} D(p,q) \Big/ (N_A \times N_B) \quad (4)$$

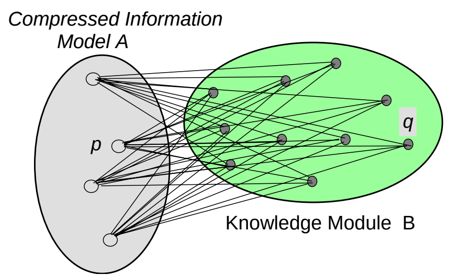
The above methods for distance compuation are also graphically illustrated in the following Fig. 3.



a) Center-of-Gravity Distance COGD between **A** and **B**



b) Minimum Distances from **A** to **B**



c) All-Pair Distances between **A** and **B**

Figure 3: Three Different Ways of Computing the Distances between two CIMs: **A** and **B**

THE METHOD FOR INFORMATION COMPRESSION

This is a two-stage procedure which first locates the neuron centers and then computes the neuron weights..

Location of the Neuron Centers by the Unsupervised Neural-Gas Learning

The aim of this computational procedure is to achieve the "most appropriate" location of the neurons in the *K*-dimensional space. A natural way to do this is to use the information about the density distribution of all available data points $\mathbf{x}_s, s=1,2,...,M$ in the space.

Basically, the learning methods in this case belong to the group of the competitive *unsupervised learning* methods and algorithms (Bishop 2003; Kohonen 2002). Among them are the well known and widely used clustering algorithms (Pedrycz 2005; Bishop 2003), the Self-Organized (*Kohonen*) Maps (Kohonen 2002), Neural Gas Algorithm (Martinetz et al. 1993) etc. All these are "data-driven" methods that tend to locate the neurons in the densest area of data in the input space.

Further on in this paper we use a modification of the popular Neural-Gas unsupervised learning algorithm (Martinetz et al. 1993) with some minor algorithmic improvements and simplifications.

<u>Initialization and Learning Parameters:</u> There are some parameters that have to be set in advance, before the unsupervised learning starts. First of all a preliminary fixed number of *N* neurons ($N \ll M$) is assumed and their initial locations (centers) are set randomly in the *K*-dimensional space:, as follows:

$$\mathbf{c}_i^{\,0} = [c_{i1}^{\,0}, c_{i2}^{\,0}, ..., c_{iK}^{\,0}], \; i=1,2,...,N.$$

The maximum number of iterations *T* for the unsupervised learning is also fixed in advance (usually between several hundreds and several thousands).

The gradually decreasing *Learning Rate* $R(t),\; 0 \le R(t) \le 1,\; t=0,1,2,...,T$ is used to control the convergence speed of the whole learning process. It is defined by the *Initial Learning Rate* R_0 and the *Steepness* T_C (Time-constant) of the learning, according to the following equation:

$$R(t) = R_0 \exp(-(t-1)/T_C), \; t=1,2,...,T \quad (5)$$

An illustration of the learning rate $R(t)$ with different assumed maximum number of iterations *T* is given in Fig. 4. In order to decrease the number of the learning parameters, further on in this paper we use the following relationship between T_C and *T*: $T_C = T/5$.

The exponentially decreasing *neighborhood function* $H_s(t,r)$ is another learning parameter that plays an important role in the Neural-Gas unsupervised learning algorithm. It is a kind of variable scaling factor for the *amount* of the update for each neuron, depending on the *relative distance* of this neuron to the current *s*-th data point $\mathbf{x}_s$ and also on the *current iteration t*.

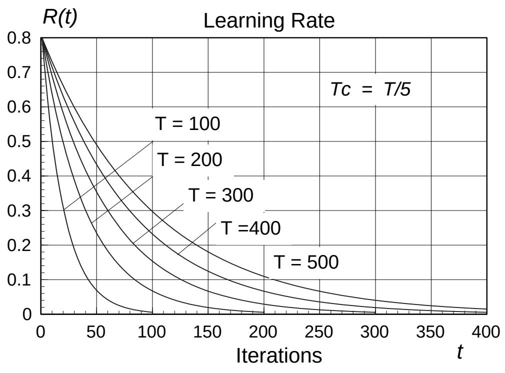


Figure 4: Illustration of the Decreasing Learning Rate *R(t)* for Different Number of Iterations

The relative distance is measured as an integer *ranking position* $r \in \{1,2,3,\ldots,N\}$ of this neuron, compared with the positions of all other neurons in the space to the same data point $\mathbf{x}_s$. Then, the neighborhood function is computed as:

$$H_s(t,r) = \exp\left(-(r-1)/B(t)\right),\ t=1,2,\ldots,T \quad (6)$$

where $B(t)$ is the current *Width* of the neighborhood function, computed as follows:

$$B(t) = \exp\left(-(t-1)/T_B\right),\ t=1,2,\ldots,T \quad (7)$$

It is seen from (7) that the width is also gradually decreasing function of the iterations. The *steepness* T_B of the width $B(t)$ controls the speed of decreasing the widths. We assumed in this paper $T_B = T_C$.

For a given data point $\mathbf{x}_s$, the ranking positions $r \in \{1,2,3,\ldots,N\}$ for all N neurons are computed through a sorting procedure in *ascending order* of the *Euclidean distances* $D_S(n)$ between all the neurons $n,\ n=1,2,\ldots,N$ and this data point:

$$D_S(n) = \sqrt{\sum_{j=1}^{K}\left(x_{sj} - c_{nj}\right)^2} \quad (8)$$

The neuron with the minimum distance to $\mathbf{x}_s$ (the nearest neuron to this data point) is called "*winning neuron*" and gets the ranking position $r=1$. The *second-nearest* neuron to the same data point gets position $r=2$ and so on until certain neuron is ranked as the last one ($r=N$) being the furthest neuron to the data point $\mathbf{x}_s$.

The following Fig. 5. illustrates the evolution of the neighborhood function $H_s(t,r)$ during learning with *T = 100* iterations.

It is seen from the figure that during the iterations, the neighborhood function (6) gradually decreases the amount of update for all the neurons, except for the "winning neuron" which gets always the full update $H_s(t,r)=1.0$, since $r=1$.

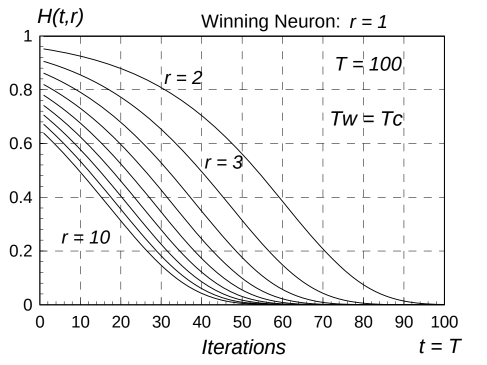


Figure 5: Evolution of the Neighborhood Function for the First *10* Neurons.

Learning Step: This step consists of a fixed number of *T* iterations: $t = 1,2,\ldots,T$ at which all available learning data: $\mathbf{x}_s,\ s=1,2,\ldots,M$ from the given learning data set are used. The purpose is to gradually move the centers $\mathbf{c}_i, i=1,2,\ldots,N$ of all N neurons into the areas in the parameter space with a bigger data density. This is done by the following general incremental learning rule:

$$\mathbf{c}_i(t) = \mathbf{c}_i(t-1) + \Delta\mathbf{c}_i(t),\ i=1,2,\ldots,N \quad (9)$$

The increment $\Delta\mathbf{c}_i(t)$ for the movement of each neuron's center is computed as:

$$\Delta\mathbf{c}_i(t) = R(t)H_s(t,r)\left[\mathbf{x}_s - \mathbf{c}_i(t-1)\right],\ i=1,2,\ldots,N \quad (10)$$

An illustration example for creating a compressed information model by using the above unsupervised learning algorithm is presented in the next Fig. 6. by using 759 normalized raw operation data from a diesel engine of a hydraulic excavator. The information from these data is approximated (compressed) by using *N = 25* neurons.

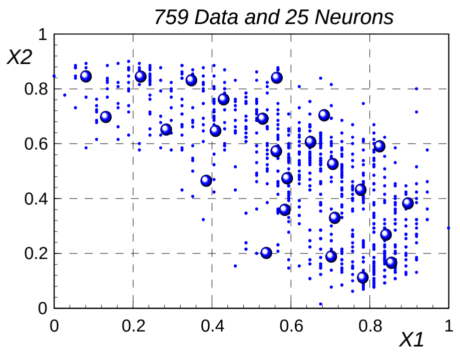


Figure 6: Compression of *759* Raw Data by using 25 Neurons

Computing the Neuron Weights G

The neuron weights $\mathbf{G}=\{g_1, g_2, \ldots, g_N\}$ are scalar values, which serve as *strength levels* for each neuron, evaluated by the number of data points that are "represented" by this neuron. In order to find this number of data for each neuron, we use again the notion of "winning neuron" from the Neural-Gas learning algorithm.

First of all, for each data point $\mathbf{x}_s$, $s=1,2,...,M$ we find the respective "winning neuron" n_s, $1 \le n_s \le N$ as the nearest neuron to this data, i.e. the neuron with a minimum Euclidean distance (8) to $\mathbf{x}_s$:

$$D_s(n_s) = \min_{1 \le n \le N} \{D_s(n)\} \qquad (11)$$

This information is further used for separating all M data into N subsets: $\mathbf{M}_n, n=1,2,...,N$, each of them containing the data points m_n, $n=1,2,...,N$ for which the same n-th neuron has been "winning neuron". In more details, the following conditions hold:

$$m_n = |\mathbf{M}_n|; 1 \le m_n \le M, n=1,2,...,N; \qquad (12)$$

$$\mathbf{M} = \bigcup_{n=1}^{N} \mathbf{M}_n; \sum_{n=1}^{N} m_n = M = |\mathbf{M}|. \qquad (13)$$

Finally, the normalized weights of the neurons are computed simply as:

$$g_n = m_n / M; \ 0 < g_n \le 1, \ n=1,2,...,N, \qquad (14)$$

Each subset of data $\mathbf{M}_n, n=1,2,...,N$ forms the so called "*Voronoi polygon*" (Martinetz et al. 1993; Kohonen 2002) in the K-dimensional space, as shown in the next Fig. 7. for the example with *25* neurons from the previous Fig. 6.

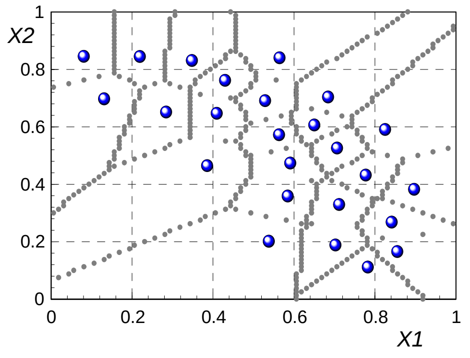


Figure 7: *25* Voronoi Polygons used for Computing the Weights of the Neurons

FAULT DIAGNOSIS OF A DIESEL ENGINE OF HYDRAULIC EXCAVATOR

The above described knowledge-based fault diagnosis method and the learning algorithms are illustrated in the sequel on the example of the performance of a diesel turbo-engine of a hydraulic excavator. All data are real, but have been normalized before their usage for the purpose of information security.

The following three parameters ($K = 3$) are considered hereafter as most important for the fault diagnosis: *P1* – Engine Speed [rpm] ; *P2* – Boost Air Pressure and *P3* – Fuel Consumption.

First of all, experimental data from the diesel engine were collected, while the excavator was working on a special proving land under different conditions, as follows: five *faulty* (abnormal) conditions, denoted as **F1,F2,…,F5** and three *normal* conditions with different load (*High*, *Medium* and *Low*), denoted as **NH**, **NM** and **NL** respectively.

The list and explanations for each operating condition are given in Table 1. The abnormal operations **F1-F5** were obtained by emulating different types of faults in the diesel turbo-engine, through special changes (deviations) in the tuning of the disel engine parts.

Table 1: List of the Preliminary Selected Operations

Code	Label	**Operation** Description
1	**F1**	Fuel Spray Nozzle deactivated
2	**F2**	Turbo-Charger Deterioration
3	**F3**	Valve Clearance Changed
4	**F4**	Air Filter Obstruction (High)
5	**F5**	Air Filter Obstruction (Low)
6	**NH**	Normal Operation (High Load)
7	**NM**	Normal Operation (Medium L.)
8	**NL**	Normal Operation (Light Load)

All *eigh*t collected data sets from Table 1. (one for each operation) consists of equal number of *1200* raw data which were used to create the Knowledge Modules in the MKB from Fig. 1.

The respective KBs were created by using the above described neural-gas algorithm with an equal number of neurons: $N = 60$ and the following learning parameters: $T = 300$; $R_0 = 0.6$; $T_C = T/5$; $T_B = T_C$.

For the purpose of the Fault Diagnosis, additional *12* data sets, named as *DS1 – DS12* were collected, each of them with number of raw data, close to *1200*. The first *8* data sets: *DS1 – DS8* were taken from the same *8* operations (*5* faulty and *3* normal), shown in Table 1., while the remaining *4* data sets: *DS9 - DS12* were taken during a real everyday working of the excavator (considered to be *Normal*) , but under slightly different working load. The respective CIMs for all data sets *DS1 – DS12* were created by the same neural-gas algorithm.

The Fault Diagnosis results, obtained by using the *Center-of-Gravity Distance* (COGD) as a "*dissimilarity degree*" are shown in the following Table 2. The first three diagnosis results are only given in this Table, arranged in a descending order of the COGD and denoted as. *No. 1*, *No.2* and *No. 3* respectively.

Table 2: Results from the Fault Diagnosis

Data. Set	Expected Fault	**Fault Diagnosis Decision**		
		No. 1	No. 2	No. 3
DS1	**F1**	**F1**	F2	F5
DS2	**F2**	**F2**	F1	F5
DS3	**F3**	**F5**	F3	NH
DS4	**F4**	**F4**	F5	F2
DS5	**F5**	**F5**	F3	NH
DS6	**NH**	**F4**	NM	F2
DS7	**NM**	**NM**	F4	F2
DS8	**NL**	**NM**	NL	F4
DS9	Normal	**NM**	NL	F4
DS10	Normal	**NL**	NM	F4
DS11	Normal	**NL**	NM	F4
DS12	Normal	**NM**	F4	F2

As seen from the Table, from all *12* operations, *9* have been properly diagnosed, while the decisions for the remaining *3* operations (the shadowed areas in the Table) were mistaken. However this misjudgment is not so big since we notice that the actual (true) fault is listed as *No. 2* candidate in two of the cases in Table 2.

Our experience showed also that the other two methods for similarity evaluation, by using the distances (3) and (4), showed slightly worse performance. Even if they properly diagnosed the cases *DS3, DS6* and *DS8,* they failed in some other cases from Table 2.

Obviously the accuracy of the fault diagnosis method depends not only on the method itself, but also on other factors, such as: *proper selection* of the number and list of parameters, used for fault diagnosis (*K = 3* used in this paper); the *real complexity* of the obtained data and also the "*quality*" of the respective trained CIM and KM in the Knowledge Base.

In order to estimate "how difficult" is this concrete problem for Fault Diagnosis, we plotted the *Centers-of-Gravity* of all *8* KMs from the Knowledge Base in Table 1. The plots for all 3 pairs of parameters: *P1-P2; P1-P3* and *P2-P3* are displayed in the next Fig. 8.

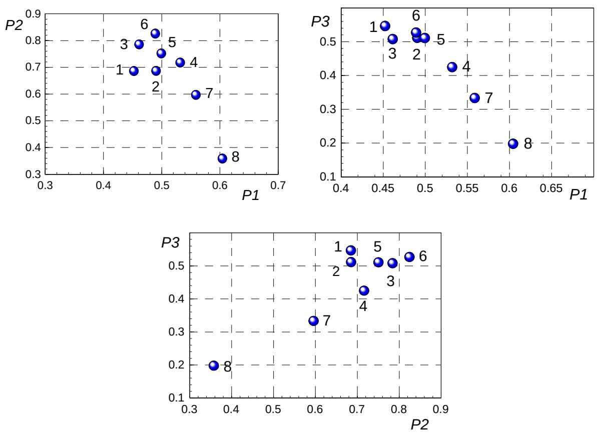


Figure 8: Locations of the Centers-of-Gravity (COG) of all *eight* Knowledge Modules (*1 – 8*) from the Knowledge Base

The plots show that most of the faulty conditions are "quite closed" to each other, which means that a big overlapping exists between the raw data sets (and the respective CIMs) of the different operations. This creates additional difficulties to the proper solution of the fault diagnosis problem and explains why some methods for similarity evaluation do not work well.

By computing the distances COGD between all-pairs of the knowledge modules and sorting them in descending order it was discovered that the closest modules (the most difficult operations to distinguish) are as follows: 1) **F3 - F5**; 2) **F2 - F5** and 3) **F1 - F5**. The pair **F6 - F8** (**NH** - **NL**) is the farthest, thus the easiest to distinguish.

CONCLUSIONS

The proposed knowledge-based fault diagnosis method in this paper uses a special technology for creating the so called compressed information models (CIM) for both the current operation data and for all modules in the knowledge base. This is an efficient and economilcal way of keeping the most representative information about any operation condition of the system. Further research in this direction is aimed at improving the quality of the compressed information models by using improved learning methods, such as growing and evolving learning, as well as at finding a more precise way for computing the similarity degree, which would further improve the fault diagnosis results.

REFERENCES

Alahakoon, D. and S.K. Halgamuge. 2000. "Dynamic Self-Organizing Maps with Controlled Growth for Knowledge Discovery", *IEEE Trans. Neural Networks,* Vol. 11, No. 3, pp. 601-614.

Bishop, C.M. 2003. *Neural Networks for Pattern Recognition*. Oxford University Press, Published in the United States.

Gertler, J. 1999. *Fault Detection and Diagnosis in Engineering Systems*, New York: Marcel Dekker.

Isermann, R. 1998. "On Fuzzy Logic Applications for Automatic Control, Supervision and Fault Diagnosis", *IEEE Trans. SMC – B*, 28, pp. 221-235.

Kohonen, T. 2002. *Self-Organizing Maps*, Third Edition, Springer Series in Information Sciences, Springer, Berlin.

Martinetz, T.; S. Berkovich and K. Schulten, 1993. "Neural-Gas Network for Vector Quantization and Its Application to Time-Series Prediction", *IEEE Trans. Neural Networks*, Vol. 4, No. 4, pp. 558-569.

Pedrycz, W. 2005. *Knowledge-Based Clustering. From Data to Information Granules*, A John Wiley & Sons Inc., Publication, Printed in USA.

Vachkov, G. 2005. "Fault Diagnosis of Complex Systems by Use of Additive Modular Knowledge Base", Proceedings of the Int. Conference on Computational Intelligence CIMCA-2005, Vienna, Austria, Nov. 28-30, IEEE Computer Society, Vol. 1, pp.1089-1094.

Zhao, Q. 2004. "Design of a Novel Knowledge-Based Fault Detection and Isolation Scheme", *IEEE Trans. SMC -B* , Vol. 34, No. 2. pp. 1089-1095.

AUTHOR BIOGRAPHY

GANCHO L. VACHKOV was born in Bulgaria in 1947. He graduated from the Technical University in Sofia, Department of Industrial Control in 1970 and obtained a Ph.D. degree in the field of Analysis of Complex Technological Systems. From 1970 until 1995 he was with the Department of Automation of Industry of the University of Chemical Technology and Metallurgy in Sofia. Since September, 1996 he has been working in Japan, in the following Universities: Kitami Institute of Technology (1996-1998); Nagoya University, Dept. of Micro-System Engineering (1998-2002) and Kagawa University (2002 – present).

The research interests and achievements of G. Vachkov are in the wide area of Computational Intelligence and Soft Computing, such as: Learning Algorithms for Fuzzy Modeling, Fuzzy and Neural Identification and Control, Intelligent Control Applications and Intelligent Data Processing for Evaluation of Systems Performance and Fault Diagnosis. The most recent applications of his research results are in the area of Maintenance and Fault Diagnosis of hydraulic excavators and other construction machines.