Appeared in *Proceedings of the 5th International Conference on Fuzzy Systems—FUZZ-IEEE'96*, volume 1, pages 206–212, New Orleans, September 1996.

A consistency checker for a fuzzy diagnosis system applied to warm rolling-mills in steelmaking plants

Wolfgang Slany*
Christian Doppler Laboratory for Expert Systems
E184/2, TU Wien, A-1040 Vienna, Austria, Europe
Phone: +43–1–58801–6123, Fax: +43–1–5055304
E-Mail: wsi@dbai.tuwien.ac.at
http://www.dbai.tuwien.ac.at/staff/slany.html

Ján Vaščák
Department of Cybernetics & Artificial Intelligence
Technical University of Košice
Letná 9, 041 20 Košice, Slovakia
Fax: +42–95–6330115
E-Mail: vascak@hron.fei.tuke.sk

Abstract

Technical diagnosis is often confronted with machines that still work but should possibly be repaired to increase efficiency or to avoid general failure. Post-mortem repair strategies are usually too late and tend to increase costs. Prophylactic maintenance of machinery at regular intervals does not lead to unexpected breakdowns but is equally costly. The best strategy would be repairing or replacing defective equipment at a certain optimal point in time usually shortly before imminent failure, the latter being in general difficult to predict correctly. However, it is possible to monitor and capture various parameters coming from the equipment through sensors and to establish some connection between these and the danger of such failure. Nevertheless, given a diagnosis problem, it is usually difficult to choose a priori parameters describing how the available data should be considered. Counterintuitive small changes in the parameters can have chaotic effects on maintenance decisions. We propose a method to select a diagnosis model corresponding to the equipment at hand. The method basically allows a consistent tuning of parameters. Some first analysis indicates that our method makes fuzzy diagnosis systems ready for the real-world.

1. Introduction

Slany [14] proposed a consistency test for knowledge-based scheduling applications that use fuzzy constraints. The consistency test allows a kind of case-based tuning of the many parameters involved in fuzzy models of such scheduling tasks. Concrete application was the fine-grained production scheduling of the LD3 steelmaking plant of VOEST Alpine Stahl in Linz, Austria.

Vaščák [15] implemented a fuzzy diagnosis system for failure detection of the transmission in a warm rolling-mill at the East Slovakian Steelmaking Works in Košice, Slovakia.

This diagnosis system is based on the vector valued fuzzy set theory of Kóczy [11] and its main task is to act as a part of a *flexible monitoring system* to replace prophylactic maintenance with a so-called *condition based maintenance*. A variety of different shades of gray, i.e., grades of damage must be used for this purpose. The use of fuzzy set theory is very useful for this multicriteria failure recognition. Further, the expert may not be able to express and describe exactly his knowledge and, as a rule, corrects his concrete parameter values in several cycles. Thus, the second reason to use fuzzy set theory is for describing such vague knowledge.

The knowledge acquisition is the greatest problem in each knowledge system. It was no exception here as well. The expert was not able to describe directly and correctly his knowledge at first. He often changed parameters and knowledge structures, and sometimes also acknowledged mistakes. Given a general diagnosis problem, it is usually difficult to choose a priori parameters describing how the available data should be considered. Slany [14] shows that counterintuitive small changes in the parameters can have chaotic effects on maintenance decisions. In light of the link between nonmonotonic logics and fuzzy partial constraint satisfaction methods as developed by Brewka et al. [3] and investigated by Guesgen [6] and Dubois et al. [4], such nonmonotonic reactions to changes in knowledge bases seem to be an obvious result. The most dangerous case of inconsistent knowledge base design en-

*Please send all correspondence regarding this paper to Wolfgang Slany.

countered in the diagnosis case was when the mistake of the expert was detected only several change cycles after the inconsistent change had taken place. The expert then had to go back and test the last cycles to find out where the consistency in his knowledge representation was broken. This problem motivated for the design of an inconsistency checker to give the expert or knowledge engineer a tool for the knowledge base development.

2. A consistency problem in the knowledge base design process

A crucial factor for the good functioning of a knowledge-based diagnosis system is the knowledge base. Therefore, the major part of the design of the knowledge system is the knowledge acquisition and the knowledge representation. The design process of the knowledge base is carried out in cycles. This design process is connected with many problems like inconsistency of knowledge, contradictions between knowledge rules, structural relations between knowledge chunks, causality problems, and representation of the knowledge.

This section deals with consistency problems arising in the process of changes in the knowledge base during such cycles. The expert changes the knowledge rules, their parameters, and relations among them in each cycle. It can easily happen that the human expert overlooks or forgets some relation, which may lead to inconsistency in the knowledge base. Slany [14] describes a constraint based consistency test that is used for fuzzy-scheduling tasks. This principle can be used for diagnostic purposes too. Diagnosis problems are not restricted to just black and white (failure or correct work conditions), but allow shades of gray (a machinery still operates but its output suffers from suboptimal working conditions, resulting in decreased output quality). Bonissone [1] describes and compares several formal methods that are well-equipped to handle such shades of gray. In order to detect a possible consistency problem in a diagnosis system's knowledge base, a tool for comparing changes at single cycles can be very helpful for the expert.

The principle consists of defining the knowledge setting in each developing cycle, where a new setting is created in each cycle. When a change is not essential, the new setting is only a fine tuning, and consistency with the old settings is not broken. In this case, the order between the two reference instantiations remains unchanged for each reference pair. If one reference ranking pair is evaluated in the opposite order, a consistency problem occurs. This means that either the new setting (configuration) is *wrong* and the expert made a mistake (e.g., overlooked some relations among production rules or set up the parameters of membership functions amiss), or that some reference ranking pairs are obsolete and should therefore be removed from the reference ranking pair database, i.e., the new design of the knowledge base is essentially changed and is not compatible anymore with the old knowledge description.

This problem is also narrowly associated with the knowledge representation in the knowledge base. Two kinds of knowledge representation, *directed structural hierarchical tree* and *matrix representation*, together with their modifications of the consistency test, are described in the following sections. These knowledge representations were used for design of a diagnostic system for failure detection of the transmission in a warm rolling-mill by Vaščák [15]. The diagnostic system was implemented in 15 man-months and is currently in use at the East Slovakian Steelmaking Works in Košice, Slovakia. We plan to extend it by the knowledge acquisition techniques introduced by Slany [14] and presented in this paper to make it easily tunable and, even more important, adaptable to the frequent changes in machinery and objectives that are encountered in present-day real-world industry applications.

2.1. The consistency test of a directed structural hierarchical tree

A knowledge representation frequently used is of the form of IF–THEN production rules. A human expert must be able to describe the knowledge in this form, i.e., the expert must be able to recognize all possible situations and predict the plausible consequences. This supposes that the expert is able to describe all parameters and relations among single production rules. So a directed structural hierarchical tree arises where its roots represent the inputs, i.e., the actual measured data and statements of the human operators in human form (values of linguistic variables). The intermediate nodes aggregates the rules, represented by the edges, that are connected. The leafs are the outputs, i.e., the results, in this case the possible failures of a transmission. A graphic representation of a directed structural hierarchical tree is given in the Figure 1, where the causal and temporal dependency between the rules is transparent. For instance, rules for diagnosing a transmission can be represented as in Table 1.

Often, the expert is not able to correctly describe the whole knowledge tree immediately at the beginning of the knowledge acquisition phase. Only a rough structure is created. At first, the parameters are only

A:	IF	The temperature increases quickly
B:	AND	the oil flows slowly
C:	THEN	the oil is maybe dirty
D:	OR	the bearing is a little bit damaged
E:	IF	The oil is maybe dirty
F:	AND	the bearing is medium old
G:	THEN	the oil is probably dirty (change it!)

Table 1. Sample rules for diagnosing a transmission.

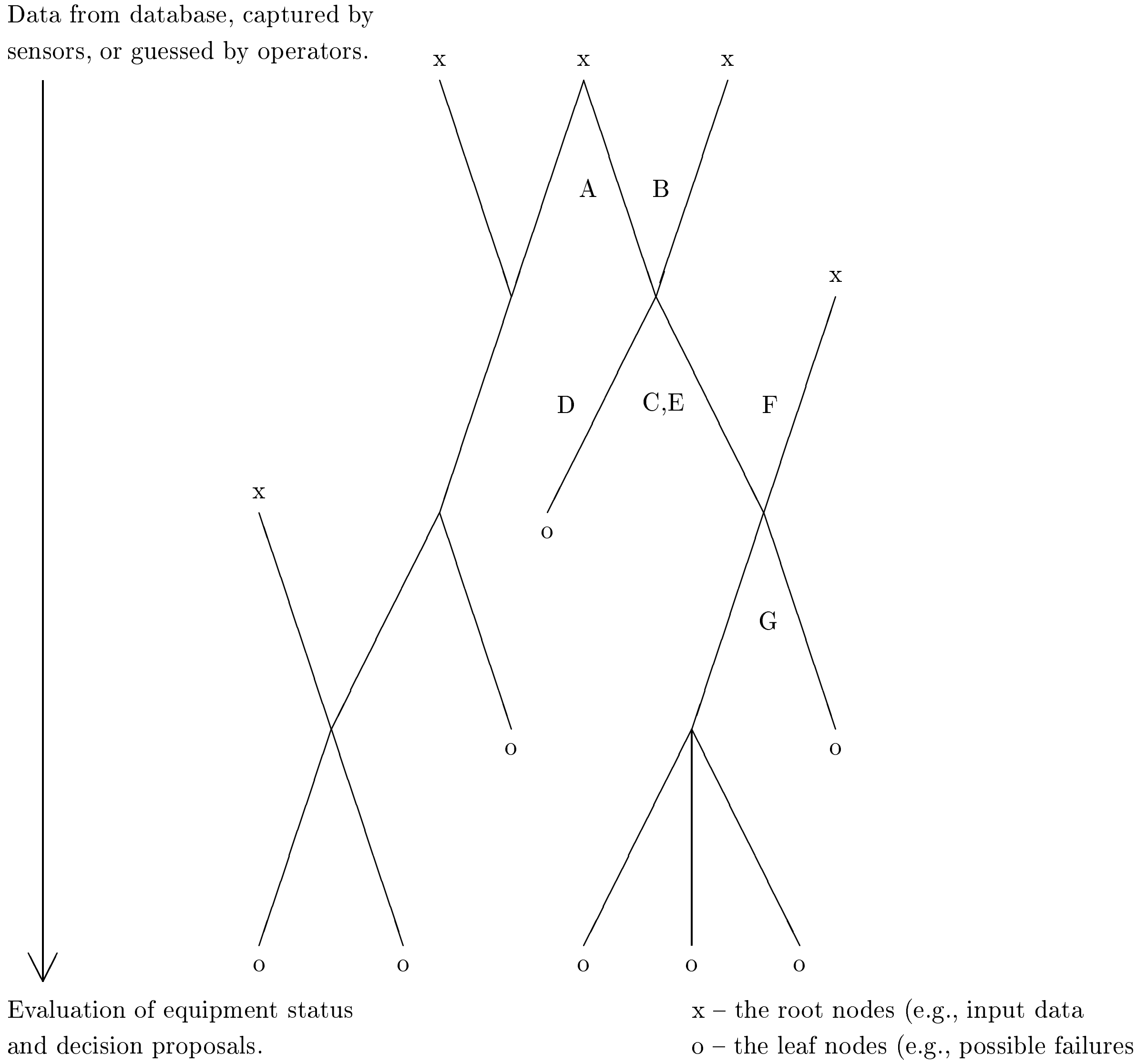

Figure 1. Graphic form of a directed structural hierarchical tree.

For $s = 1, 2, \ldots, i-1$

$$B_{i,s} = \frac{w_O + w_I + w_E}{w_O \sum_{p=1}^{n} \mid K_{O_{i,p}} - K_{O_{s,p}} \mid + w_I \sum_{r=1}^{m} \mid K_{I_{i,r}} - K_{I_{s,r}} \mid + w_E \mid K_{E_i} - K_{E_s} \mid} \quad (1)$$

where:

$B_{i,s}$	is the evaluation score value of the tree s at the configuration i;
w_O	is the chosen weight for output edges from a node;
$K_{O_{i,p}}$	is the number of nodes with p output edges at the new tree;
$K_{O_{s,p}}$	is the number of nodes with p output edges at the tree of the s-th configuration;
w_I	is the chosen weight for input edges to a node;
$K_{I_{i,r}}$	is the number of nodes with r input edges at the new tree;
$K_{I_{s,r}}$	is the number of nodes with r input edges at the tree of the s-th configuration;
w_E	is the chosen weight for number of rules (edges);
K_{E_i}	is the number of rules (edges) at the new tree;
K_{E_s}	is the number of rules (edges) at the tree of the s-th configuration.

For $s = i$ is $B_{i,i}$ an arbitrary value greater than one.

This formula uses the qualitative properties of the two tree structures for their comparison and determines in a considerable extent also the structural similarity.
The consistency condition is observed when the following inequalities are fulfilled:

$$B_{i,i} \geq B_{i,i-1} \geq B_{i,i-2} \geq \ldots \geq B_{i,1} \quad (2)$$

Table 2. A possible evaluation score function.

approximated. The domain is successively widened and the parameters are refined in further cycles. The process of the knowledge base design is similar to the one in the scheduling process. A list of possible actions is available and the result of the scheduling process is an ordered set of these actions. The ordering of these actions is represented by instantiations. The directed structural hierarchical tree is an ordering too. Its items can also be described in form of a list, and the instantiations are pointers that order these items in form of a tree. The directed structural hierarchical tree can be regarded as a schedule too. This analogy enables us to use the consistency test methodology of the scheduling process in the design of the diagnosis knowledge base as well. The expert thus makes changes in the tree, and, therefore, the question arises whether the proposed changes are compatible with the old structure of the knowledge base. In case inconsistency is detected, a warning is issued, implying that the expert maybe has made a mistake by overlooking some important aspect of the diagnosis problem. The expert can immediately retract the proposed change after analyzing the inconsistency. Without the check, the mistake could linger undetected for many more change cycles, degrading maintenance quality and possibly even leading to a deadlock situation at an unexpected moment in the future when another change is proposed. Alternatively, in case the expert is sure that the proposed change is correct and complete, then the consistency test gives a warning that the new knowledge structure is not compatible with the old structure. In this case, a human expert can be supported in developing several decision making strategies of his own in a what-if game-like setting. Different decision making strategies could be for instance 'risky/cost-cutting', 'highest-quality', 'standard-mix', etc. As the economic situation of a company changes, the priorities between antagonistic objectives such as quality versus timeliness can undergo major revisions. Therefore, having different, mutually inconsistent sets of configurations makes perfect sense in a real-world environment, as long as one does not use them in combination and one is aware of their different biases.

The possible settings (configurations) are in this diagnostic case the membership functions of the linguistic variable values, causal relations among assumptions and consequences, the actions to be carried out, importance (priorities) of the diagnostic quantities, limits, logical structure of constraints, weights of nodes and edges, logical relations among rules, and choice of the aggregation operators. The aim of all changes is to decrease the number of wrong diagnoses and to increase the number of correct diagnoses. But this condition is not sufficient. Entering another set of diagnostic cases can cause total different results of correctness of the knowledge system. Therefore, a consistency criterion is also to be investigated.

The new, changed knowledge structure is tested in each cycle. Its quality is described by *evaluation scores* that must always be functionally dependent on actual configurations in order to calculate the new evaluation scores for all the old ordered pairs of instantiations. There are many possible evaluation score functions. A reasonable way to design an evaluation score function is through the similarity between the old and the new tree structure. A possible evaluation score function can be described as in Table 2.

Human experts can specify implicitly the overall configuration of the parameters by asserting a set of 'normal' reference database entries. The easiest way to apply the heuristic that establishes consistent configuration parameters is to let the human expert do parameter changes, and to later check them out with the introduced consistency test.

2.2. The consistency test of a matrix knowledge representation

The human expert is sometimes not able to describe the knowledge in a clearly structured form of rules. This can be caused by very vague and intuitive knowledge. A matrix knowledge representation form is more suited for this case. The expert needs not exactly describe all relations in the domain. Only an approximate specification of the matrix elements is necessary, similar to the methods developed by Saaty [13] for the analytic hierarchy process. Obviously, the tuning phase has as a rule more tuning cycles than in the previous case, since the specification of the parameters is often performed rather aleatorily.

In general, the matrix knowledge representation form is described by a set of *alternatives* $A_1, A_2, \ldots, A_n$ and a set of *criteria* $E_1, E_2, \ldots, E_m$. Thus, a matrix M_i of dimensions $[n \times m]$ (Table 3) arises where each element e_{ij} is in general a fuzzy relation between alternative A_i and criteria E_j.

The total evaluation of an alternative A_i is described by a fuzzy vector as defined by Kóczy [11]:

$$\overline{e_i} = (e_{i1}, e_{i2}, \ldots, e_{im}) \tag{3}$$

The meaning of the evaluation depends on the concrete application, e.g., the possibility of failure arising or the extent of damage in the system to maintain. The goal

$A_i \backslash E_j$	E_1	E_2	$\cdots$	E_j	$\cdots$	E_m
A_1	e_{11}	e_{12}	$\cdots$	e_{1j}	$\cdots$	e_{1m}
A_2	e_{21}	e_{22}	$\cdots$	e_{2j}	$\cdots$	e_{2m}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
A_i	e_{i1}	e_{i2}	$\cdots$	e_{ij}	$\cdots$	e_{im}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
A_n	e_{n1}	e_{n2}	$\cdots$	e_{nj}	$\cdots$	e_{nm}

Table 3. The matrix knowledge representation.

of the inference mechanism is to evaluate vector (3) and to get the defuzzified evaluation of a plausible alternative in scalar form. This is done by a *composition function*. Often the chosen composition function needs a matrix W_i of weights w_{ij} (for each e_{ij}) in order to take into account the differing importances of influence factors E_j on alternatives A_i. To calculate the evaluation, it is for instance possible to compute the weighted average as in

$$e_i = \frac{\sum_{j=1}^{m} w_{ij} \, |e_{ij}|}{\sum_{j=1}^{m} w_{ij}} \tag{4}$$

where $|e_{ij}|$ are the defuzzified instantiated membership functions e_{ij}.

Alternatively, other aggregation schemes such as *t*-norms [9] could be used. However, to achieve more human-like decision making, softer aggregation operators having nice mathematical properties have been proposed, for instance by Yager [16]. Another interesting averaging aggregation method, which in contrast to most others is also associative, was proposed by Klement et al. [10] and successfully applied to an application similar to the one described in this paper by Bonner et al. [2]. We have so far used the aggregation scheme of (4) successfully to deal with the diagnosis problem, but are planning to compare it to other methods soon.

The knowledge representation form can be described also by production rules independent of each other. The evaluations are the assumption parts of a rule, and each alternative is a consequence. Thus, the number of rules is equal to the number of alternatives. In other words, a set of n one-edged trees arises.

The fuzzified matrix knowledge representation is defined such that the elements e_{ij} are not scalar values but membership functions. The composition function is an arbitrary aggregation operator. The possible settings (configurations) are in this case the parameters of membership functions, the weights, and the choice of a composition operator. A new knowledge matrix M_i and weight matrix W_i are created in each cycle i. The evaluation score $B_{i,s}$ between two matrixes M_i and M_s can be calculated in two phases:

1. The similarity between a membership functions $e_{i_{ij}}$ from M_i and $e_{s_{ij}}$ from M_s can be calculated using the principle of the geometric similarity of the membership functions. Various definitions of metrics can be used for this purpose, e.g., the definitions by Minkowski [7], Euklides, Hausdorf [12], Pomer [8], or Hamming. The result of the first phase is the similarity matrix $\underline{P}_{i,s} = [p_{ij}]$ containing scalar values.

2. The total similarity, i.e., evaluation score $B_{i,s}$, can be determined by an arithmetic average

$$B_{i,s} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}}{n\,m} \tag{5}$$

The procedure of the consistency test is performed in a similar way as the one described in Section 2.1. Again, consistency condition (2) must be fulfilled.

3. Conclusion

Slany [14] proposed a consistency test for knowledge-based scheduling applications with fuzzy constraints. The consistency test allows a kind of case-based tuning of the many parameters involved in fuzzy models of such scheduling tasks. Concrete application was the fine-grained production scheduling of the LD3 steel-making plant of VOEST Alpine Stahl in Linz, Austria.

Vaščák [15] implemented a fuzzy diagnosis system for failure detection of the transmission in a warm rolling-mill at the East Slovakian Steelmaking Works in Košice, Slovakia.

In the present paper, we describe how to reuse ideas from the consistency test of the scheduling application [14] and to apply this test to the diagnosis problem [15]. The basic idea is simple, in that a set of earlier *characteristic* changes in parameters are stored in a database, together with explanations regarding the changes provided by human experts, such that these stored changes in a certain sense constrain future changes. This not only allows us to tune parameters of the diagnosis system, but also to easily adapt the diagosis system to modifications of equipment, sensors, or priority shifts in the production objectives. Another application of the consistency test is that, based on actual cases, several different experts are supported in reaching agreement on parameters of a knowledge base for real-world decision making. Further, even a single expert can develop several decision making strategies of his own in a what-if game-like setting, such as for instance 'risky/cost-cutting', 'highest-quality', 'standard-mix', etc.

While such a consistency checker's performance can be difficult to assess because the working conditions change a lot during its life-time and results get lost in plethora of competing efforts to increase overall efficiency, we are currently investigating a more formal assessment of the consistency test's efficiency by statistical means over a longer period of time.

References

[1] P. P. Bonissone. Reasoning, plausible. In Stuart C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 2, pages 1307–1322. John Wiley & Sons, Inc., 2nd enlarged and revised edition, 1992.

[2] M. Bonner, S. Mayer, A. Raggl, and W. Slany. FLIP++: A fuzzy logic inference processor library. In *Workshop Notes of the IJCAI'95 Workshop on Fuzzy Logic in Artificial Intelligence*, pages 83–92, Montreal, Canada, August 1995. URL: ftp://ftp.dbai.tuwien.ac.at/pub/papers/slany/ijcai95 wsflai1.ps.gz.

[3] G. Brewka, H. W. Guesgen, and J. Hertzberg. Constraint relaxation and nonmonotonic reasoning. Technical Report TR-92-002, International Computer Science Institute (ICSI), Berkeley, Calif., 1992. URL: ftp://ftp.icsi.Berkeley.edu/pub/techreports/1992/tr-92-002.ps.Z.

[4] D. Dubois, H. Fargier, and H. Prade. Propagation and satisfaction of flexible constraints. In R. R. Yager and L. A. Zadeh, editors, *Fuzzy Sets, Neural Networks and Soft Computing*, pages 166–187. Van Nostrand Reinhold, New York, 1994.

[5] D. Dubois, H. Prade, and R. R. Yager, editors. *Readings in Fuzzy Sets for Intelligent Systems.* Morgan Kaufmann, 1993.

[6] H. W. Guesgen. A formal framework for weak constraint satisfaction based on fuzzy sets. In *Proceedings of ANZIIS-94*, pages 199–203, Brisbane, Australia, 1994. URL: ftp://ftp.icsi.Berkeley.edu/pub/-techreports/1994/tr-94-026.ps.Z.

[7] A. Kaufmann. *Introduction to the Theory of Fuzzy Subsets*, volume 1 of *Fundamental Theoretical Elements.* Academic Press, New York, 1975.

[8] A. Kaufmann and M. M. Gupta. *Introduction to Fuzzy Arithmetics.* Van Nostrand Reinhold, New York, 1985.

[9] E. P. Klement. Operations on fuzzy sets and fuzzy numbers related to triangular norms. In *Proceedings of the Eleventh International Symposium on Multiple-Valued Logic*, pages 218–225, Norman, 1981. IEEE, New York.

[10] E. P. Klement, R. Mesiar, and E. Pap. Associative compensatory operators. In Mario Fedrizzi, Erich P. Klement, Aldo Ventre, and Alessandro Zorat, editors, *Proceedings of CIFT'94: Current Issues in Fuzzy Technologies: Decision Models and Systems*, Trento, Italy, June 1994.

[11] L.T. Kóczy. Vectorial I-fuzzy sets. In M. M. Gupta and E. Sanchez, editors, *Approximate Reasoning in Decision Analysis*, pages 151–156. North-Holland, Amsterdam, 1982.

[12] A. L. Ralescu and D. A. Ralescu. Probability and fuzziness. *Inform. Science*, 34:85–92, 1984.

[13] T. L. Saaty. *The Analytic Hierarchy Process.* McGraw-Hill, 1980.

[14] W. Slany. Scheduling as a fuzzy multiple criteria optimization problem. *Fuzzy Sets and Systems*, 78:197–222, Mar. 1996. Issue 2. Special Issue on Fuzzy Multiple Criteria Decision Making; URL: ftp://ftp.dbai.tuwien.ac.at/pub/papers/slany/fss96.ps.gz.

[15] J. Vaščák. *The use of fuzzy set theory in technical multicriteria diagnosis.* PhD thesis, Košice, Slovakia, FEI TU, 1996. In Slovak language.

[16] R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems, Man, Cybernetics*, 18:183–190, 1988. Republished in Dubois et al. [5, pp. 80–87].