Selected Problems of Expert Systems for Technical Diagnostics

Wojciech CHOLEWA

Silesian Technical University, Gliwice, Poland

Summary: Knowledge representation plays an important role in all AI systems. Most of all expert systems make use of rules. For applications in technical diagnostics it is necessary to consider a large amount of domain expertise. Troubles turn up for persons that try to modify or maintain the existing rule-basis. The paper shows different approaches to decision tables, which are useful in the process of knowledge acquisition from experts with high skill in technical diagnostics and with little experience in knowledge engineering.. The discussion starts with well-known, simple decision tables. It introduces the notions of multi-layer as well as uncertain tables. They can be interpreted as a particular form of knowledge representation. Particular focus is set on such extensions to simple decision tables that allow to represent the necessary and sufficient conditions in an uniform and easy way. An additional aim of the paper is to point out the particular usefulness of blackboard structures in expert systems for technical diagnostics.

1. Introduction

The efficiency of modern machinery increases with availability. Therefore the main goal of any plant engineer is to decrease the time unavoidably necessary to repair and/or exchange worn out parts and increase the time intervals between failures. The strategy for reliable plant operation can be defined as the condition monitoring of normal and faulty states of a technical processes on the basis of measured features of interaction between the process and environment, to aid or yield an automatic decision on the process performance. One of the advancing way of state recognition is to use the vibrations immanently generated by running machinery. Recent measuring techniques enable to observe the great amount of features of diagnostic signals. Successful vibration measurement and analysis require an intimate familiarity with types of measurement, transducer characteristics and application, plus the capabilities and limitations of the diagnostic instrumentation.

Resulting features are often compared with warning, pre-alert and alert levels to detect whether observed quantities overcome pre-set limit values. Other, more sophisticated methods for machine monitoring and failure diagnostics are required to ascertain safe and economic operation of complex machinery as well as the quality control of products. Among a great number of more conventional techniques of failure detection and condition monitoring applied to technical diagnostics, the development of expert systems, which aims at emulating the way utilised by human experts for solving

problems, plays a fundamental role. The basic elements of an expert system are shown in Fig. 1. It can significantly improve the symptom interpretation and the system modelling in case of malfunctions or algorithmically difficult to describe systems and processes.

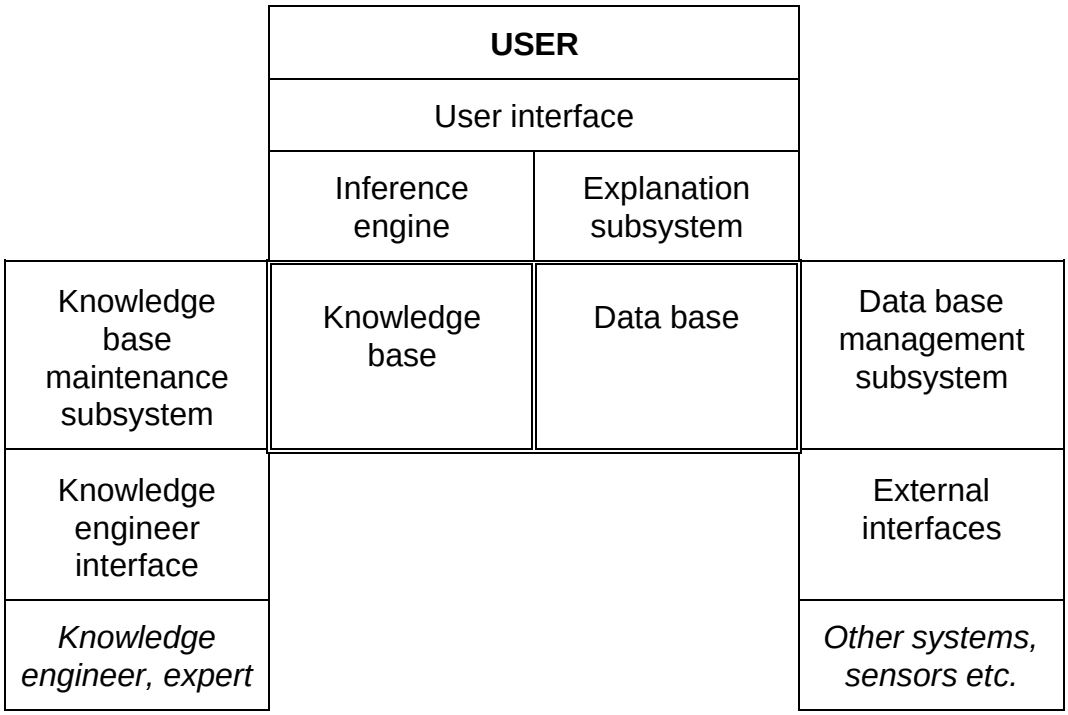

Fig. 1. Basic elements of an expert system.

2. Knowledge representation

A critical step in the development of an expert system is the acquisition of expert knowledge. Most of all expert systems make use of rules (and structures basing on rules) which are able to represent nearly all kinds of knowledge. They allow to include additional information about the strategy of reasoning which improves the inference mechanism. Many authors point out that rule-based systems are easy to understand by experts and allow to acquire new knowledge easily. It is true, that the direct acquisition of rules from experts or books is possible, but it may be connected with serious difficulties.

For applications in selected domains (for instance in technical diagnostics) it is necessary to consider a large amount of domain expertise in the rule-basis. The difficulties result from relationships that exist between rules but are not known or are not clear to knowledge engineers and experts. Troubles turn up for persons that try to modify or maintain the existing rule-basis. It seems that an important cause of the difficulties is the lack of an appropriate ordering of rules into small, well defined contexts. It is evident that such ordering can be accomplished by means of hierarchical structures and object-oriented approaches. Promising results follow from the application of decision tables, too. Such tables allow to isolate base subsets of rules and are a very useful tool for an interface between the knowledge engineer and experts.

2.1. Simple decision table

A basic worksheet for the decision table (Fig. 2) consists of rows $c \in [1,C]$ representing conditions and rows $a \in [1,A]$ representing actions and/or conclusions. It is possible to define several different conditions and actions for each rule. The left-hand side of the table contains definitions of premises or tasks returning premises $C(1),...,C(c),...,C(C)$ and definitions of actions $A(1),...,A(a),...,A(A)$. The columns on the right-hand side of the table contain the definition of the rules $r \in [1,R]$. The entries $C(r,c)$ of conditions for these rules consist of expected Boolean values of premises (answers to questions or test results). Legal entries $C(r,c)$ for each condition $C(c)$ in each rule $R(r)$ are:

- Y means the expected value of the premise = YES,
- N means the expected value of the premise = NO,
- - means that the premise is not included in the current condition.

Legal entries $A(r,a)$ for each action $A(a)$ in each rule $R(r)$ are:

- X means *execute*,
- blank means *do not execute*.

Sometimes the rows $E(e)$ representing exits $E(r,e)$ are included in the table too (Fig. 3). It is possible to define several different exits, but only one exit in each rule.

Topic:		**Rules**				
		1	**...**	***r***	**...**	***R***
1	***C*(1)**	***C*(1,1)**	**...**	***C*(*r*,1)**	**...**	***C*(*R*,1)**
...	**...**	**...**	**...**	**...**	**...**	**...**
c	***C*(*c*)**	***C*(1,*c*)**	**...**	***C*(*r*,*c*)**	**...**	***C*(*R*,*c*)**
...	**...**	**...**	**...**	**...**	**...**	**...**
C	***C*(*C*)**	***C*(1,*C*)**	**...**	***C*(*r*,*C*)**	**...**	***C*(*R*,*C*)**
1	***A*(1)**	***A*(1,1)**	**...**	***A*(*r*,1)**	**...**	***A*(*R*,1)**
...	**...**	**...**	**...**	**...**	**...**	**...**
a	***A*(*a*)**	***A*(1,*a*)**	**...**	***A*(*r*,*a*)**	**...**	***A*(*R*,*a*)**
...	**...**	**...**	**...**	**...**	**...**	**...**
A	***A*(*A*)**	***A*(1,*A*)**	**...**	***A*(*r*,*A*)**	**...**	***A*(*R*,*A*)**

Fig. 2. Simple decision table worksheet (after [8] with slight modifications).

The order of tasks carried out in the table may be explained by means of the table cursor. It has to be assumed, that:

- rules are tested from left to right, i.e. the cursor proceeds through the rules from $R(1)$ to $R(R)$,

- entries of conditions in the table are tested for each rule in top-down order, i.e. the cursor proceeds from the condition $C(r,1)$ to $C(r,C)$,
- values of premises $C(c)$ are inspected (called) only when they are used in current entries of the conditions $C(r,c)$,
- the value of each premise $C(c)$ is calculated (estimated, results from the response to a question) and additional tasks necessary to establish the values are carried out at the first call for such a value in the table only (the value remains unchanged for all the next calls),
- each not fulfilled condition $C(r,c)$ results in a jump to the next rule, i.e. the cursor proceeds to the rule $R(r+1)$,
- if all the conditions for the tested rule $R(r)$ are fulfilled, then the rule is fired and the actions $A(r,a)$ associated with the rule are initiated in top-down order,
- if all the actions associated with the fired rule are done or there are no fired rules in the table then the cursor exits from the table.

Sleeve Bearing		**Rules**						
		R1	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**
C1	Self-excited vortex vibration with a frequency of about one-half of the rotating speed of the shaft.	**Y**	**N**	**-**	**N**	**N**	**Y**	**Y**
C2	Temperature of the oil is too high.	**Y**	**Y**	**Y**	**N**	**-**	**-**	**N**
C3	Temperature of the oil is too low.	**-**	**-**	**Y**	**N**	**Y**	**Y**	**N**
C4	Bearing has a lemon-shaped working surface of the sleeve.	**-**	**-**	**-**	**-**	**-**	**-**	**N**
A1	Radial clearance ought to be reduced.						**X**	
A2	Radial clearance ought to be increased.		**X**					
A3	Amount of oil that must be pumped through the bearing ought to be decreased.					**X**		
A4	Thermal calculations of the bearing ought to be performed.	**X**						
A5	Lemon-shaped working surface of the sleeve ought to be introduced.							**X**
E1	Error. Clear and repeat the table.			**X**				
E2	Quit the table.	**X**	**X**		**X**	**X**	**X**	**X**

Fig. 3. Example of a decision table.

It means that not all the rules will be tested and not all the premises have to be called. The fixed way for testing the conditions (from left to right / top-down) determine the

order of the called premises $C(c)$ and the order of questions forwarded to the users. This property may be useful for the user-friendly design of dialog processes. There are several interesting sorting and optimising criteria [8] for decision tables which can be taken into account. For a decision table to be completed each possible combination of entries for the given premises has to be included once and only once. Missing rules, redundant rules and conflicting rules are not admitted. To obtain a complete decision table (i.e. a table without missing rules) it may be assumed that if the cursor reaches a rule with an empty section of conditions, the rule is fired. Such a rule should be placed as the last rule in the table (because other rules following this one are not accessible).

2.2. Multi-layer decision table

A large decision table is hard to handle. It is difficult to display (or print) all the columns at the same time and the designer of the table can lose a general view on the dependencies between rules as well as on interactions between the premises. It is highly inconvenient to check the completeness and consistency of rules.

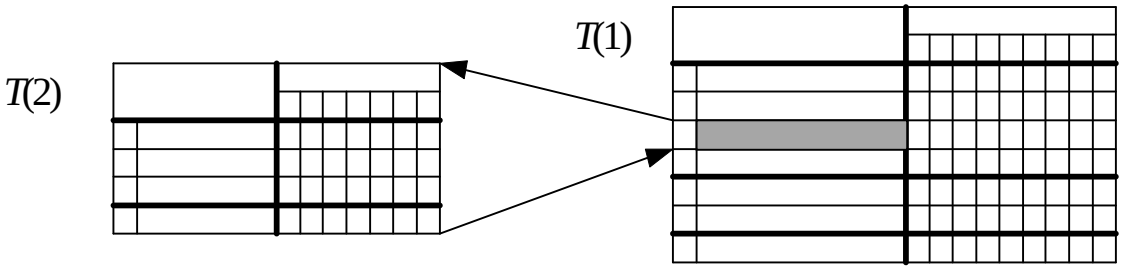

Fig. 4. Multi-layer decision table.

Topic:		**Rules**					
		1	**2**	**...**	***r***	**...**	***R***
1							
...							
C							
1							
...							
A							
V		***V*(1)**	***V*(2)**	**...**	***V*(*r*)**	**...**	***V*(*R*)**

Fig. 5. Worksheet for a decision table which returns a value.

It is often possible to select a small subset of rules and their premises. They may be clustered and moved into a separate decision table. The question is how to establish a link between the tables or how to include such a small table as a sub-table in the main one. The simplest way (Fig. 4) is to put this table $T(2)$, as a special case of a new

premise, into the main table $T(1)$. To do this it is necessary to assign return values $V(1),..., V(r),..., V(R)$ to all the columns on the right-hand side of the sub-table. They can be easily defined by an additional bottom row (labelled *V*) in the worksheet presented in Fig. 2 (see Fig. 5). This row should contain values assigned to the table, adequate to the fired rules. Moreover, a default value $V(0)$ ought to be assigned to the whole table in the case of tables with missing rules.

3. Statements

Tentative applications of expert systems which ought to contain a large amount of domain expertise that are not given in a rigorous form (e.g. technical diagnostics), result in conclusions that we have to deal with rules which are true in most, but not in all, cases. It means that the statements and rules in such systems are often uncertain. We will not distinguish the different reasons of uncertainty (random, unprecise, approximate ones) in this paper. Uncertain statements and rules can be represented in a lot of different ways, where certain rules and certain statements can always be taken into account as a special case of uncertain ones.

3.1. Facts and statements

We need some information about the real word to reason in an expert system. Such information can be presented in the form of statements, called sentences and most often facts. Of course, from the direct use of the notion *fact* a lot of misinterpretations may result. Statements describing the real world express only the particular point of view (e.g. *Mary is beautiful*) or belief that something has happened (will happen) or has been (will be) done (e.g. *it will be raining*). Statements depend on current (or default) contexts. They are not independent, real existing facts.

We make the following distinguishing:

- fact - exists in the real world, independent from our perception,
- statement - a record of our opinion about selected facts.

We can record our opinion about the given fact by means of different statements (*Mary is beautiful*, *Mary is ugly*). The measure expressing our belief that the particular statement is true will be called *the value of the statement*. The value of the statement should be distinguished from *the contents of the statements*.

The above formalisation is necessary for the clear discussion concerning uncertainty in expert systems (which follows).

3.2. Contents of statements

Statements are applied to describe the properties, peculiarities and features of real and abstract objects. They may be written down by means of the following triples

$$Statement = < Object, Attribute, Value > \quad (1)$$

stating that to the designated *Object* (e.g. oil in the journal bearing #2) belongs the defined *Attribute* (e.g. mean oil temperature) of the given *Value* (e.g. 55 °C).

The *Object* may be defined as an element belonging to the family of all subsets of so-called *primary objects* (e.g. parts of machine elements or even fragments of these parts).

Such an assumption makes it possible to regard interactions between elements of machine in an uniform way as special kind of (abstract) objects.

To simplify the reasoning process we ought to assume that *Objects, Attributes* and even *Values* are elements of corresponding (finite) dictionaries

$$Object \in DictionaryOfObjects \tag{2}$$

$$Attribute \in DictionaryOfAttributes \tag{3}$$

$$Value \in DictionaryOfValues \tag{4}$$

where *DictionaryOfAttributes* and *DictionaryOfValues* are defined for classes of objects and not for their instances.

3.3. Value of statements

The simplest approach is to consider the values $v(s)$ of statements s as logical values

$$v(s) \in \{ \text{YES, NO} \}. \tag{5}$$

Diagnostic knowledge is mostly acquired from experts. We are often forced to apply rules that are regarded to be true in the most cases (but not always). Hence, rules and statements implying conclusions may be uncertain and/or imprecise. Several empirical and theoretically based, ad hoc approaches representing and recording approximate statements are known (see e.g. [7], [13], [12], [9], [3], [10]). The simplest approach is the direct application of the probability theory and standard Bayesian model. A modification of the probability theory results in the *truth values* $T(s)$ from the range [0, 1], assigned to each statement s.

$$v(s) = T(s) \in [\,0, 1\,]. \tag{6}$$

They are interpreted as an extension of two logical values NO=0 and YES=1, onto the ordered set [0, 1] of real numbers. Particular implementations differ mainly in the interpretation of the value $T(s)$=0, which can point at statements that are false or which can point at only those statements for which we have not got any source of information that they are true. The last case does not mean that there do exist any reasons to interpret such statements as false.

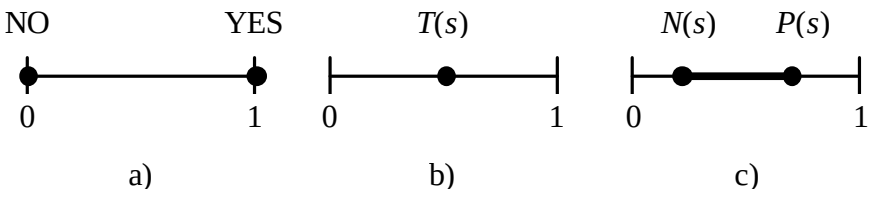

Fig. 6. Values of statements: a) Boolean, b) truth value, c) necessity and possibility.

An interesting extension of the truth values results from modal logic. The notions of possibility and necessity form concept for the measures of *possibility* $P(s)$ and *necessity* $N(s)$ assigned to the statements s [6], [7]. Apart from a rigorous explanation, we may interpret the values of these measures

$$[\,N(s), P(s)] \subset [\,0, 1\,]. \tag{7}$$

as boundaries of a hypothetical range of the unknown truth value (Fig. 6):

$$0 \le N(s) \le T(s) \le P(s) \le 1 \tag{8}$$

By means of $N(s)$ and $P(s)$ we can distinguish the case of compensated premises *pro* and *contra* $N(s)=P(s)=0.5$ from the case with a lack of information $N(s)=0$ and $P(s)=1$. It may be useful to draw the necessity and possibility in a special plot (Fig. 7), where the value of the statement s is presented as a segment (range) from $N(s)$ to $P(s)$. The segment contains unknown truth values of the statement. The plot (Fig. 7) contains the limits $N_{\min}$ and $P_{\max}$ for $N(s)$ and $P(s)$, too. We are able to assign a particular meaning to selected regions of the plot (Fig. 8).

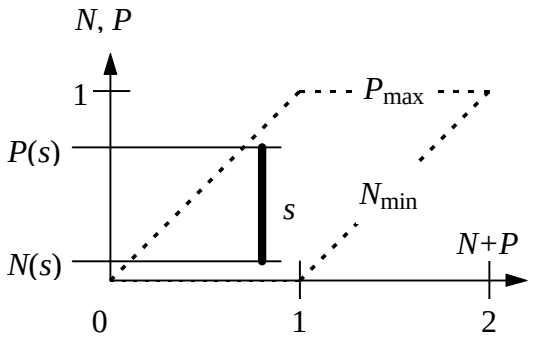

Fig. 7. Plot of an uncertain statement *s*.

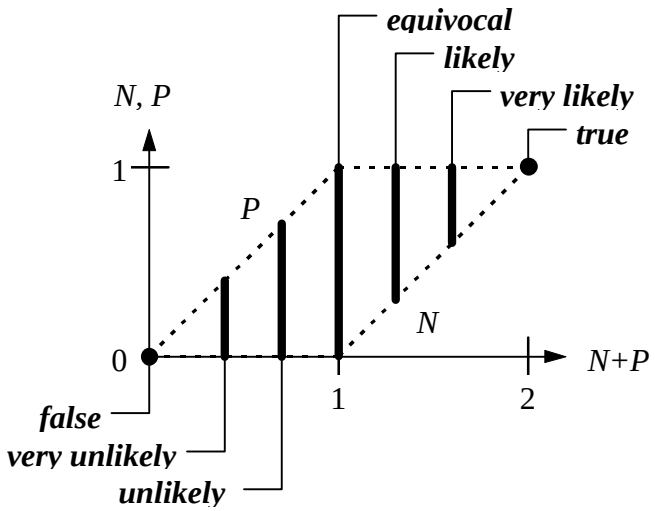

Fig. 8. Linguistic descriptions (values) of uncertain statements (example).

Neglecting details it is worth to stress that limited exactness of the rules makes it possible to limit the degree of accuracy of considered attribute value [3]. It means that instead of *quantitative* values (e.g. 55 °C) the *qualitative* values (e.g. low) may be used. If we decide to apply only *classes of values* instead of *accurate values* (preferring qualitative values) we need to use only finite set of considered values (4), which may be pointed by their names.

4. Rules

The main assumption for all rule-based expert systems is the existence of links (relations) between the statements. By means of such links represented as rules we can draw conclusions about new values of statements which result from the known values of other statements. Many authors express the opinion that the rules in a rule-base represent causal links between the statements. This is often not true. It is clear that statements should be connected by links representing their cross-dependencies, but such dependencies need not be causal. For example, the statements p and q in (9) as well as r and s in (10) form pairs of linked statements (but the dependencies are not causal):

$$\underline{\text{if}}\ x\ \underline{\text{then}}\ p\ ; \qquad \underline{\text{if}}\ x\ \underline{\text{then}}\ q\ ; \tag{9}$$

$$\underline{\text{if}}\ r\ \underline{\text{then}}\ y\ ; \qquad \underline{\text{if}}\ s\ \underline{\text{then}}\ y\ ; \tag{10}$$

How to represent links between statements?

4.1. Continuous implication

Mathematical logic introduces the notion of implication. If p and q are given statements, then the implication $p \rightarrow q$ is true when q is true or p is false. Classical logic uses two basic patterns for the deduction in propositional calculus, called *modus ponens* (11) and *modus tollens* (12):

if the rule (implication)	$p \rightarrow q$	is true	
and the premise	p	is true	(11)
then the conclusion	q	is true	

if the rule (implication)	$p \rightarrow q$	is true	
and the premise	$\neg q$	is true (i.e. q is false)	(12)
then the conclusion	$\neg p$	is true (i.e. p is false)	

It is important to point out, that:

- if we know only that p is false (or respectively, that q is true) we are not able to arrange a reliable inference about the logical value of q (or respectively p),
- we are able to draw reasonable conclusions only when the implication $p \rightarrow q$ is true, because from the assumption that the implication is false it follows only that p has to be true and q has to be false.

For reasoning by means of truth values we need an extension of implication. A two-valued implication is shown in Fig. 9.a. To extend the definition into a definition of continuous implications which are needed to deal with the truth values $T(s) \in [0,1]$ we have to define a surface (a function) spanned on the points shown in Fig. 9.a. An interesting example of continuous implication is Łukasiewicz's implication (Fig. 9.b):

$$T(p \rightarrow q) = \min\left(1 - T(p) + T(q),\ 1\right) \tag{13}$$

On the basis of (13) or similar continuous implications we can generalise the modus ponens and modus tollens (see e.g. [7]).

4.2. Bilateral implication

It has been mentioned that we often have no reasons to assume that the relation represented by a rule in a rule-base is a causal relation. How to model the causality and how to model the lack of causality? One way is to write the relations in such a form that both modus ponens and modus tollens may be applied together. This can be done by means of bilateral implication. The bilateral implication $p \leftrightarrow q$ is simply a pair of both underlying implications $p \rightarrow q$ and $q \rightarrow p$. The bilateral implication has to be symmetric for modus ponens and modus tollens to make the result of reasoning independent of the particular forms of statements (*object X has the property A* and *object X has the property* $\neg A$) and questions sent to the user.

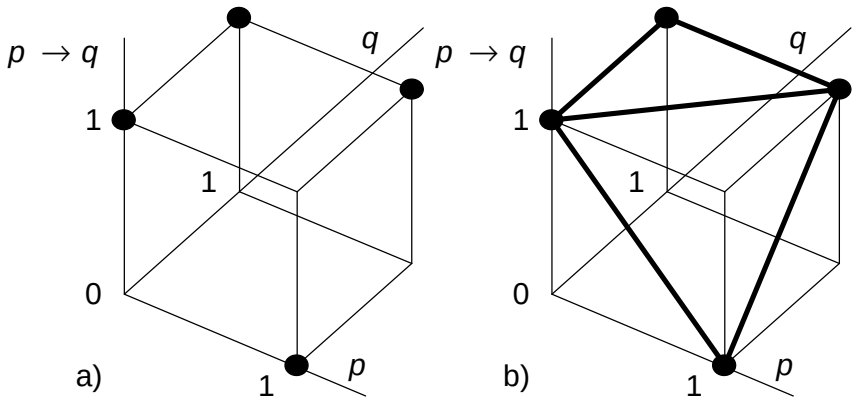

Fig. 9. Implication: a) two-valued, b) continuous (13).

The notion of necessity and possibility and the concept of bilateral implication can be used together. They result in the following joint (generalised) modus ponens and modus tollens:

$$\begin{array}{l} T(p \rightarrow q) \\ T(q \rightarrow p) \\ \hline N(p),\, P(p) \quad \text{such that} \quad N(p) \le T(p) \le P(p) \\ N(q),\, P(q) \quad \text{such that} \quad N(q) \le T(q) \le P(q) \end{array} \tag{14}$$

where for known values $T(p \rightarrow q)$, $T(q \rightarrow p)$, $N(p)$, $P(p)$ and for continuous implication (13) we have

$$\begin{aligned} N(q) &= \max\left(0,\ N(p) + T(p \rightarrow q) - 1\right) \le N(p) \\ P(q) &= \min\left(1,\ P(p) - T(q \rightarrow p) + 1\right) \ge P(p) \end{aligned} \tag{15}$$

Bilateral implication allows to write rules containing both the necessary and sufficient conditions, in an uniform way. Bilateral implication can be presented in the plots of necessity and possibility (e.g. Fig. 10). Such plots allow to interpret the results of calculations carried out by means of (15).

Bilateral implication allows to consider sufficient as well as necessary conditions. For example, let us consider the following statements:

$$\begin{cases} x\text{: } 'K \text{ is a multiple of 2'}, \\ y\text{: } 'K \text{ is a multiple of 4'}, \\ z\text{: } 'K \text{ is a multiple of 8'}. \end{cases} \tag{16}$$

where

- the statement *y* is a sufficient condition for the statement *x*,
- the statement *y* is a necessary condition for the statement *z*,

or in equivalent forms

$$T(y \to x) = 1; \quad T(x \to y) = 0 \tag{17}$$

$$T(y \to z) = 0; \quad T(z \to y) = 1 \tag{18}$$

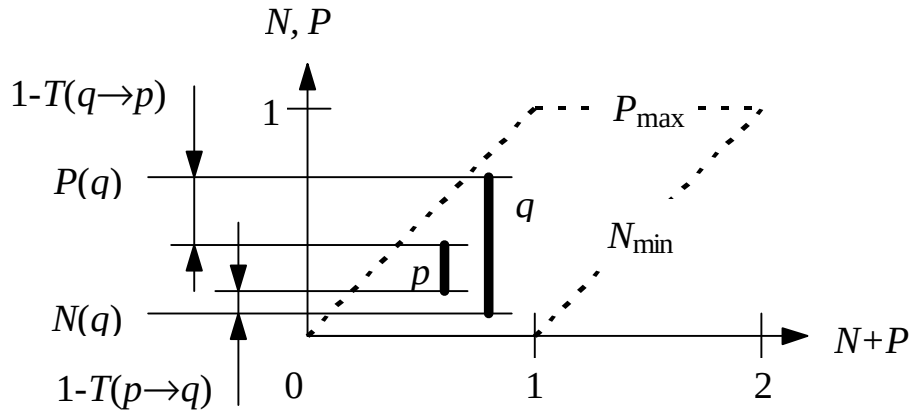

Fig. 10. Bilateral implication (14), (15).

From the plot in Fig. 11 it follows that

- if *y='likely'* then *x='very likely'* and *z='equivocal or unknown'*,
- if *y='unlikely'* then *x='equivocal or unknown'* and *z='very unlikely'*.

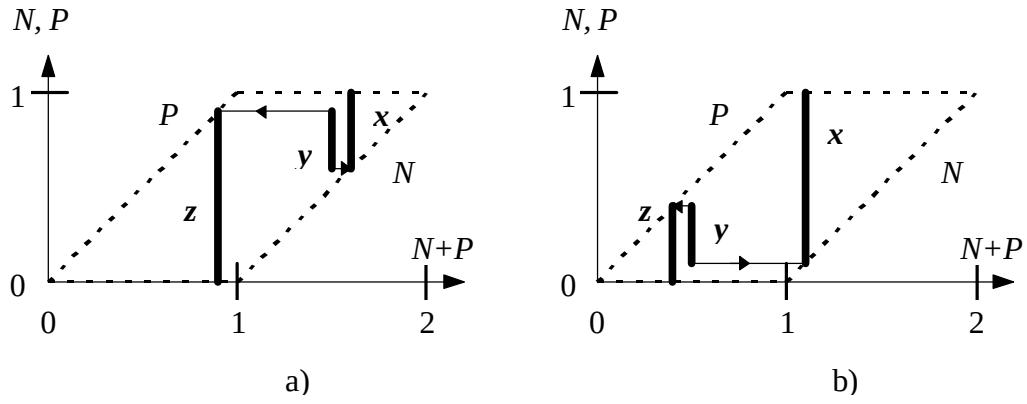

Fig. 11. Examples of (17) and (18) for: a) *y*='likely', b) *y*='unlikely'.

5. Uncertain decision tables

The notions of bilateral implication, necessity and possibility can be used to define uncertain decision tables forming an extension of the idea of Boolean (sharp) decision tables. In order to introduce the notion of an uncertain decision table we assume that each rule $R(r)$ in the table may be interpreted in a dual form

$$\begin{cases} \text{if } \mathit{conditions}(r) \text{ then } \mathit{actions}(r) \\ \text{if } \mathit{conditions}(r) \text{ then } \mathit{hypothesis}(r) \end{cases} \tag{19}$$

where the *hypothesis*(r) is an abstract element used only to explain of the idea of uncertain tables. This element will be called a hidden hypothesis of the rule $R(r)$.

Basic assumptions about the elements of simple decision tables (see Chapter 2.1) and about multi-layer decision tables (see Chapter 2.2) still hold true for uncertain tables. They should be extended by the following ones:

- values of premises $C(c)$ are returned as pairs of numbers $(N(c),P(c))$ representing the measures of necessity and possibility for true values of the premise,
- entries of conditions $C(r,c)$ are written for each premise c and each rule r, as pairs of numbers $(T(c\rightarrow r),T(r\rightarrow c))$ representing true values $T(\mathit{premise}\rightarrow\mathit{hypothesis})$ and $T(\mathit{hypothesis}\rightarrow\mathit{premise})$ of a bilateral implication,
- testing the condition $C(r,c)$ results in the conditional necessity $N(r|c)$ and conditional possibility $P(r|c)$ of hidden hypothesis, calculated by means of (15):

$$\begin{aligned} N(r|c) &= \max\left(\, 0,\ N(c) + T(c \rightarrow r) - 1 \,\right) \\ P(r|c) &= \min\left(\, 1,\ P(c) - T(r \rightarrow c) + 1 \,\right) \end{aligned} \tag{20}$$

- testing all the conditions for the rule r results in the necessity $N(r)$ and possibility $P(r)$ of hidden hypothesis, common to all the conditions of the rule r and calculated as follows

$$\begin{aligned} N(r) &= \max_{c=1,\ldots,C}\left(N(r|c)\right) \\ P(r) &= \min_{c=1,\ldots,C}\left(P(r|c)\right) \end{aligned} \tag{21}$$

- the additional top row for each rule r contains a pair of numbers specifying the limits $N_{\min}(r)$ and $P_{\max}(r)$ for the necessity $N(r)$ and possibility $P(r)$ of hidden hypothesis,
- the rule r is fired if and only if the specified limits will not be exceeded by the necessity and possibility of hidden hypothesis which result from the conditions, i.e.

$$N_{\min}(r) \le N(r) \le P(r) \le P_{\max}(r) \tag{22}$$

- legal entries $A(r,a)$ for each action are:

  Y indicates an action which should be carried out if the rule is fired,

  N indicates an action which should be carried out if the current, inspected rule is not fired,

X indicates an action which should be carried out for the current rule, independent of the results of conditions,

blank indicates an action which should not be carried out,

- the bottom row for each rule contains necessity and possibility which form the return value of the fired rule.

From (20) and (21) it follows that it is possible to obtain the following result

$$N(r) > P(r) \tag{23}$$

which indicates contradictory premises $C(c)$ or inconsistent conditions $C(r,c)$. Due to (22) all the rules with conditions resulting in (23) will newer be fired.

5.1. Dialog with an expert

In order to use the uncertain decision table we have to acquire the values $(T(c\rightarrow r),T(r\rightarrow c))$ from experts. Of course, the direct use of numbers may be a bit confusing. More user-friendly is the use of the linguistic descriptions of selected, particular values of bilateral implication $(T(c\rightarrow r),T(r\rightarrow c))$. For example, we can introduce the following set of values (with their acronyms and descriptions), that are easy to understand by the users

- (1.0, 0.0) ≡ SUF ≡ sufficient condition,
- (0.8, 0.0) ≡ RSUF ≡ rather sufficient condition,
- (1.0, 1.0) ≡ EQ ≡ equivalent statements,
- (0.0, 0.8) ≡ RNEC ≡ rather necessary condition,
- (0.0, 1.0) ≡ NEC ≡ necessary condition.

The above acronyms may be written directly as $C(r,c)$ values in the decision table.

6. Message-based data management

Conclusions about the current technical state of an object result from the data about the object. The amount of useful (varying with time) data may be extremely large and the designing of an appropriate data base for diagnostic expert system is not an easy task [11]. In general it is obvious to design the logical and physical structure of the data base, independent from any particular application. Existing interfaces, e.g. ODBC (Open Database Connectivity) permit an interoperability and allow for a single application an access to data in different DBMS (Database Management Systems). A simple and most reliable solution is the application of relational data bases consisting of related sets (tables) of records (rows) containing fields (columns). Existing DBMS allow to run queries asking questions about the data stored in data base and returning filtered subsets of data. Such queries may be created e.g. by means of SQL (Structured Query Language) statements.

Managing a large data base is a complex operation. Moreover the acquisition of diagnostic data is, by its nature, a continuous process. Taking into account the above remarks as well as the scope of diagnostic tasks it is convenient to make use of the concept basing on *blackboard* structure.

6.1. Blackboard

The blackboard provides the basic functionality for data acquisition and reasoning processes. It forms a view attached to a data base and acts as an intermediary between the data base and the user or tasks participating in the acquisition and reasoning.

Under blackboard (Fig. 12) we understand an (abstract) collection of messages that are accessible to their receivers (C). Messages are delivered by units (A) to the blackboard manager (B) that is responsible for

- propagation of messages containing news in the format suitable for insertion of these messages into the blackboard,
- insertion of these messages on the blackboard,
- removal from the blackboard these messages that are out of date.

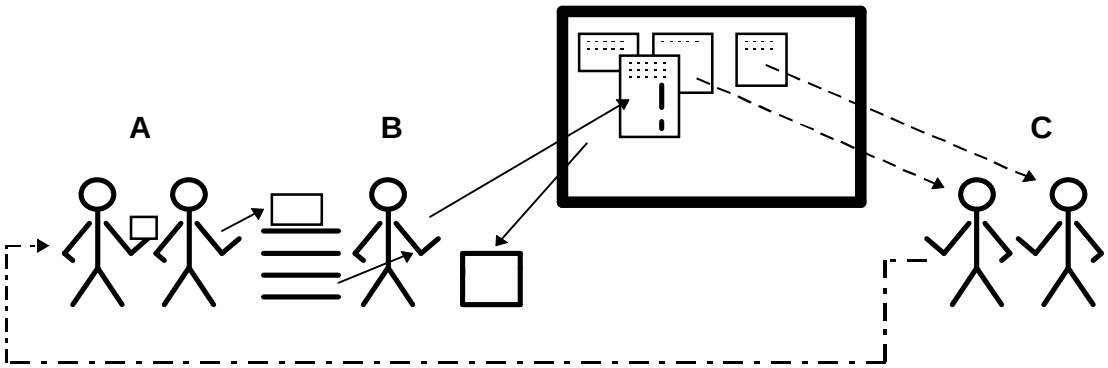

Fig. 12. Blackboard-like diagnostic system.

Receivers of messages (C) basing on the contents of messages can submit new messages to a manager (B). It is possible to consider systems with

- single blackboard or several blackboards (forming several *discussion groups* - e.g. 'short-term problems', 'long-term problems'),
- single manager or several managers (with differed levels of significance and/or rights),
- a set of agents (multi-agent system) where several applications (agents) work upon the same problem and then their results are joined with making use of aggregation of agents opinions if it were necessary.

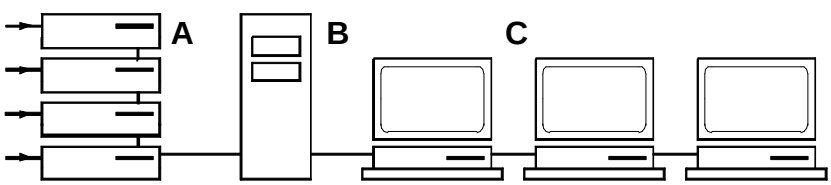

Fig. 13. Exemplification of a blackboard.

If we consider diagnostic systems for large and complex machinery then a network of computers may forms a simple exemplification of blackboards with a moderate investment and exploitation costs. Here it is possible to recognise (Fig. 13)

- computers running as drivers/interfaces (A) and co-operating directly with measuring devices that are sources of messages (Fig. 14),
- workstations (C) that enable users of the system to conduct dialogues with the diagnostic system,
- the selected computer (B) which contains an appropriate DBMS and which is capable to serve as the blackboard manager.

<table>
<tr><td colspan="3">Message</td></tr>
<tr><td colspan="3">Network interface</td></tr>
<tr><td colspan="2">On-line multi-channel data acquisition unit
Slow and fast analogue-digital converters
Synchronisers
Real time digital signal processing</td><td rowspan="3">Off-line

Data collector</td></tr>
<tr><td>Transducers
Displacement transducers
Proximity probes
Velocity probes
Accelerometers
Key phasors, Tachometers</td><td>Process variables
Temperature
Load
Pressure
Flow</td></tr>
<tr><td colspan="2">Machinery</td></tr>
</table>

Fig. 14. Selected sources of messages.

The discussed network makes it possible to share tasks and resources as well as allows for parallel processing on several computers in the network.

6.2. Messages, news and statements

The blackboard simplifies the management of messages. What does it mean *message* (Fig. 15)?

Statement (contents of the statement) is a kind of pronouncement that does not contain information whether this proclamation is true or false. *News* consists of statement contents and statement values. Assigning *truth value* (Fig. 6) to the *statement* (1) we obtain *news* that may be regarded as predicates in the predicate calculus

$$News = < Statement, TruthValue > \qquad (24)$$

News is a substance of a *message* that also contains information on the source and destination of the *News* and information about validity conditions represented most often by time interval, written in general form *'from* t_i *to* t_j*'* or in peculiar forms *'since t'*, *'to t'*,

'always'. Using validity condition of the *message* it is possible to define corresponding strategy of *forgetting* of out-to-date messages, which may be performed either as deleting of selected messages or transferring (moving) such messages to external data bases.

We assume that placing the given *message* (Fig. 15) onto the *blackboard* (Fig. 12) is equivalent to the supposition that corresponding statement takes the pointed-out truth value. Hence lack of the given message on the blackboard is not equivalent to falseness of the corresponding statement.

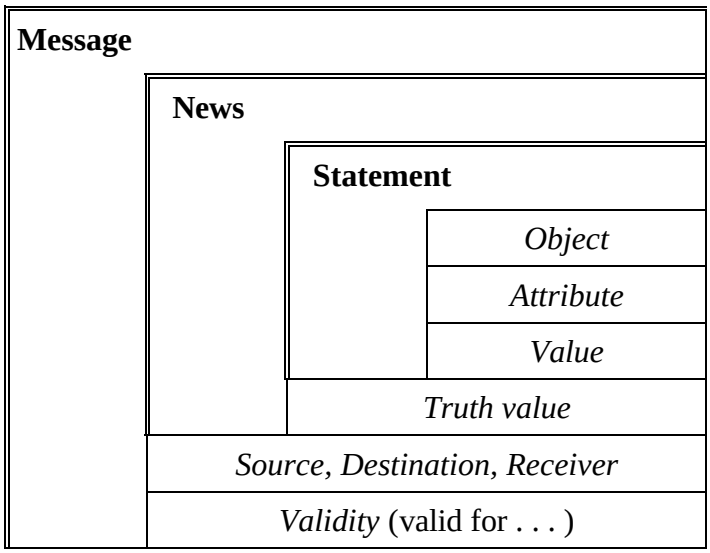

Fig. 15. Structure of a message.

If we decide to apply only classes of values (4) of statements in (1) instead of accurate values we need only a finite set of statements considered in the system. Hence we are able to define a set of standard statements (called *DictionaryOfStatements*) by means of (2), (3), (4)

$$DictionaryOfStatements \subset DictionaryOfObjects \times DictionaryOfAttributes \times DictionaryOfValues \tag{25}$$

DictionaryOfStatements (25) gives us the possibility to apply *news* (24) containing only identifiers *StatementID* of standard news (e.g. order numbers or names)

$$News = < StatementID, TruthValue > \tag{26}$$

This simplifies the degree of complexity of activities undertaken by the blackboard manager and makes easier the preparation of the knowledge base. Next step towards simplicity is connected with limited set of *truth values* (see e.g. Fig. 8) of statements that belong to finite *DictionaryOfStatements* and results in a finite *DictionaryOfNews*

$$DictionaryOfNews \subset DictionaryOfStatements \times DictionaryOfTruthValues \tag{27}$$

The reasoning / concluding process based upon messages containing standard news (these belonging to the *DictionaryOfNews*) takes place in the so-called *closed world*, hence the standard news may be the only result of concluding.

7. An expert system shell

MAS (Maintenance Aid Shell [2], [4], [5]) is an expert system shell containing the production system, frame interpreter, frame editor, browser/debugger, reference data base and an interface for extracting diagnostically useful statements from external data bases. The interface isolates the actual knowledge base of MAS from any particular machinery data base.

MAS runs under MS Windows. The frame interpreter of MAS is a processing unit specially designed to handle different types of statement structures, represented by means of frames. It can also handle uncertain statements and co-operate in real time with other reasoning systems and data sources (e.g. drivers of measuring instrumentation) by means of DDE. LISP-like frame description language allows to take into account different kinds of inheritance of frames. It enables us to control the degree of encapsulation. This is achieved by the use of demons (making no difference between data and a description of data, i.e. a code). All the elements of frames are identified by their names. Names need not to be globally unique. We can use the same name for slots in different frames. This results in a polymorphism, i.e. the names are shared and their meanings depend on the given context. Frame structures of MAS are open for modifications by the running program. It means that it is possible for the running program to change the algorithm for current tasks.

MAS is capable to interpret uncertain decision tables. Tables can be included in the source code for a frame-interpreter. It is possible to use external decision tables by means of ODBC or DDE (e.g. from the running MS Access). The table returning a value can be used as a premise in another table. It is useful to write such a table as a special case of a frame. The premises of the table ought to be written as frames, too. Such frames allow to simulate some kinds of backward chaining, and to write rules dealing with the domain knowledge as well as with meta-knowledge about the reasoning process in a similar, uniform way.

8. Conclusion

Different kinds of decision tables and particularly multi-layer, uncertain decision table form a convenient tool for interactions between the knowledge engineer and experts in the process of the acquisition of knowledge for expert systems with forward chaining [1]. The tables can be used in expert systems containing an appropriate table-interpreter. It is very useful to use the blackboard structure to organise a message-based reasoning in expert systems.

References

[1] Cholewa W.: *Uncertain decision tables as a tool for knowledge acquisition*. Second International Workshop on Learning in Intelligent Manufacturing Systems. Budapest 1995, p.628-654.

[2] Cholewa W.: *Shell Expert System MAS. User's Manual*. KPKM Gliwice, 1993. Report RMT6100 (manuscript, in Polish).

[3] Cholewa W., KaŸmierczak J.: *Data Processing and Reasoning in Technical Diagnostics*. WNT, Warszawa 1995.

[4] Cholewa W., Moczulski W.: *Problems of Knowledge Representation and Acquisition for the Expert System Shell MAS*. Intelligent Information Systems. Practical Aspects of Artificial Intelligence II. Inst. of Comp. Science PAS, Warszawa 1993, p.299-314.

[5] Cholewa W., Moczulski W., Filipowicz M.: *Knowledge-based Diagnosing of Technical State of Machinery*. International Workshop on Intelligent Information Systems. Wigry 1994. Proceedings p.148-161.

[6] Cholewa W., Pedrycz W.: *Expert Systems* (in Polish). Textbook No. 1447. Politechnika Œl¹ska, Gliwice 1987.

[7] Dubois D., Prade H.: *Possibility Theory. An Approach to Computerised Processing of Uncertainty*. Plenum Press, New York 1988.

[8] Hurley R.B.: *Decision Tables in Software Engineering*. Van Nostrand Reinhold, New York 1983.

[9] Kruse R., Meyer K.D.: *Statistics with Vague Data*. Reidel Publ.Comp., Braunschweig 1987.

[10] Moczulski W.: *Problems of Knowledge Acquisition for Diagnostic Expert Systems.* Proceedings of the XIII IMEKO World Congress 1994, vol.2, p.1224-27.

[11] *Reliability Data Banks*. Cannon A.G., Bendell A. (eds.) Elsevier, London 1991.

[12] Shortliffe E.H.: *Computer-Based Medical Consultation MYCIN*. Elsevier, New York 1976.

[13] Zadeh L.A.: *The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems.* Elsevier Science Publishers, North-Holland 1983.