KNOWLEDGE-BASED EXPLANATION IN MULTI-ATTRIBUTE DECISION MAKING *

Marko Bohanec(1) Vladislav Rajkovič(2,1)

(1) 'Jožef Stefan' Institute, Jamova 39, YU-61111 Ljubljana, Slovenia
(2) University of Maribor, Faculty of Organizational Sciences, Kranj

Abstract: An approach to decision making that combines multi-attribute decision making techniques with expert systems is described. In this approach, knowledge about a particular decision making problem is represented in the form of tree-structured criteria and decision rules. Decision making is supported by an expert system shell. This chapter focuses on the part of the shell that supports evaluation, explanation and analysis of alternatives. The corresponding algorithms are presented and discussed.

1 Introduction

In general, the *decision making problem* can be defined as follows:

Given a set of *alternatives* $\mathcal{A} = \{a_1, a_2, \ldots\}$ and (somehow expressed) *aims* or *goals* of the decision maker,

find alternative $a_i \in \mathcal{A}$ that best satisfies the goals.

Problems of this kind can be found in almost any field of human activity, ranging from everyday personal decisions to more complex problems in economy, management, medicine, etc. The complexity of such problems usually originates in:

- complex and often incomplete, uncertain or conflicting knowledge of how to define and achieve the goals,
- numerous and/or loosely defined alternatives,
- a large number of parameters that influence the decision,
- the presence of several decision making groups with different objectives, and
- time constraints and other resource limitations imposed on the decision making process.

A number of methods and computer programs have been developed to *support* decision makers in solving more or less complex decision problems (Humphreys and Wisudha 1987). They are usually

*A reformatted version of the chapter published in *Computer aided decision analysis: Theory and applications* (ed. Nagel, S.), 189–204, Quorum Books, 1993.

studied within the framework of decision support systems (Keen and Scott Morton 1978, Alter 1980), operations research and management sciences, decision theory (French 1986) or decision analysis (Phillips 1986). In a broad sense, *expert systems*, which originate in artificial intelligence, can also be considered systems that solve and explain complex repetitive decisions (Efstathiou and Mamdani 1986; Turban 1988; Klein and Methlie 1990).

In this chapter we show how multi-attribute decision making methods can be *combined* with expert systems, in particular how to support *explanation* and *analysis* of decisions. The concept of multi-attribute decision making is presented in section 2. Section 3 gives a formal description of knowledge representation that is used in the proposed approach. Methods and algorithms for the evaluation, explanation and analysis of alternatives are presented in sections 4, 5 and 6. Finally, practical applications of this approach are summarized in section 7 and discussed in section 8.

2 Multi-attribute decision making

In *multi-attribute* decision making, the decision problem is decomposed into smaller, less complex subproblems (Keeney and Raiffa 1976; Chankong and Haimes 1983; French 1986). Such subproblems are represented by a set of *criteria* (or *attributes*) $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$. Criteria are variables that take values from the corresponding value domains $X_1, X_2, \ldots, X_n$.

Each alternative $a \in \mathcal{A}$ is measured and described by a vector of values

$$a = \langle v_1, v_2, \ldots, v_n \rangle, \quad v_i \in X_i, \quad i = 1, 2, \ldots, n.$$

The alternative is then evaluated, usually in two steps. First, the decision maker's *preferences* p_i of a according to each separate criterion x_i are obtained:

$$p_i = f_i(v_i), \quad i = 1, 2, \ldots, n.$$

Here, f_i is a *partial utility function* defined by the decision maker for each x_i. The preferences p_i are usually expressed as numbers on a certain interval, for example $[0, 1]$.

The *total utility* of a is finally aggregated by the function F:

$$u(a) = F(p_1, p_2, \ldots, p_n).$$

In practice, F is most commonly expressed as a weighted sum where weights w_i are defined by the decision maker:

$$F(p_1, p_2, \ldots, p_n) = \sum_{i=1}^{n} w_i p_i.$$

The alternatives are finally ranked according to the total utilities. In general, the alternative with the highest utility should be selected as the best one.

Note, however, that only the main ideas of multi-attribute decision making have been presented above; the details are treated quite differently in different methods. For example, criteria may be additionally structured into trees. There are also considerable variations in the representation and assessment of alternatives and utility functions (French 1986; Humphreys and Wisudha 1987).

The most common drawback of existing multi-attribute methods, at least for some classes of problems, is the need of translating the decision makers' *knowledge* about a decision problem into numbers and functions. Experience with expert systems (Turban 1988; Klein and Methlie 1990) indicates that this is not always necessary. There are decision problems in which *qualitative judgement* prevails over more or less exact quantitative evaluation. For such problems, it is quite a natural choice to use models that incorporate qualitative elements as well, for example qualitative (descriptive, linguistic, ordinal) variables and rules. Expert system applications have confirmed that such an approach is not only feasible, but can also provide some valuable features: explicit and flexible knowledge representation (Fox 1985), transparency (Lévine *et al.* 1990), natural language dialogues (Vari and Vecsenyi 1988) and explanation of reasoning (Efstathiou and Mamdani 1986).

In the following sections we show how *explanation* facilities can be gained by combining multi-attribute and expert system paradigms. The proposed approach is based on an explicit articulation of *knowledge* about a particular decision making problem. Knowledge representation is specific and oriented toward multi-attribute problems. It consists of tree-structured criteria and utility functions which are represented by rules rather than formulas.

3 Knowledge representation

Knowledge representation for multi-attribute decision making is based on a *tree of criteria* $T = (\mathcal{X}, S)$, where:

$\mathcal{X}$ is a set of *criteria* $\{x_1, x_2, \ldots, x_n\}$; each criterion x_i may take values from the corresponding value *domain* X_i.

S is a mapping from $\mathcal{X}$ to the power set of $\mathcal{X}$. For each criterion, S defines the set of its immediate descendants (sons) in the tree. Only such mappings S are permitted which structure criteria into a single (connected) tree whose root is x_1.

Criteria x for which $S(x) = \emptyset$ are *leaves* of the tree. They are also referred to as *basic criteria.* The remaining ones are called *aggregate criteria.*

Domains $X_1, X_2, \ldots, X_n$ are assumed to be discrete and finite. They generally consist of words, not numbers; words such as *good* or *low* can be used. Numerical quantities can be expressed as intervals of values; each interval is treated as a discrete domain element. It is also recommended to order domain elements by preference from "bad" (non-preferred) to "good" (highly preferred) values, for example {*bad, acceptable, good, excellent*} or, for price, {*high, medium, low*}.

The tree of criteria represents the structure of a particular decision problem. The purpose of the second component of the decision knowledge base, *utility functions*, is to define the relation between aggregate criteria and their descendants in the tree. For each aggregate criterion x_k, the corresponding utility function

$$F_k : X_{k_1} \times X_{k_2} \times \cdots \times X_{k_m} \longrightarrow X_k$$

should be defined by the decision maker. Here, $X_{k_1}, \ldots, X_{k_m}$ denote the value domains of all the descendants of x_k.

When different decision groups with different interests are involved in the decision making process, each group may define their own set of utility functions. Therefore, more than one function can be defined for each aggregate criterion, as shown in figure 1.

Utility functions F_k are represented by *elementary decision rules* of the form

$$\textbf{if } x_{k_1} = v_{k_1} \textbf{ and } \ldots \textbf{ and } x_{k_m} = v_{k_m} \textbf{ then } x_k = v_k,$$

where v_{k_i} and v_k denote single values taken from domains X_{k_i} and X_k, respectively.

The consistency of rules is required. This means that the rules should be unique with respect to their conditional parts. On the other hand, the completeness of rules is not required, meaning that for some combinations of values $v_{k_1}, \ldots, v_{k_m}$ there might be no corresponding rules. The set of rules that correspond to a particular aggregate criterion is usually represented in a tabular form (see, for example, table 2).

After the criteria tree and utility functions have been defined, the evaluation of alternatives can start. The alternatives are first measured and described by values of basic criteria. The utility functions are then applied in a bottom-up manner in order to obtain aggregate values for each alternative. These values, particularly the one that has been assigned to the root, are finally used to select the best alternative.

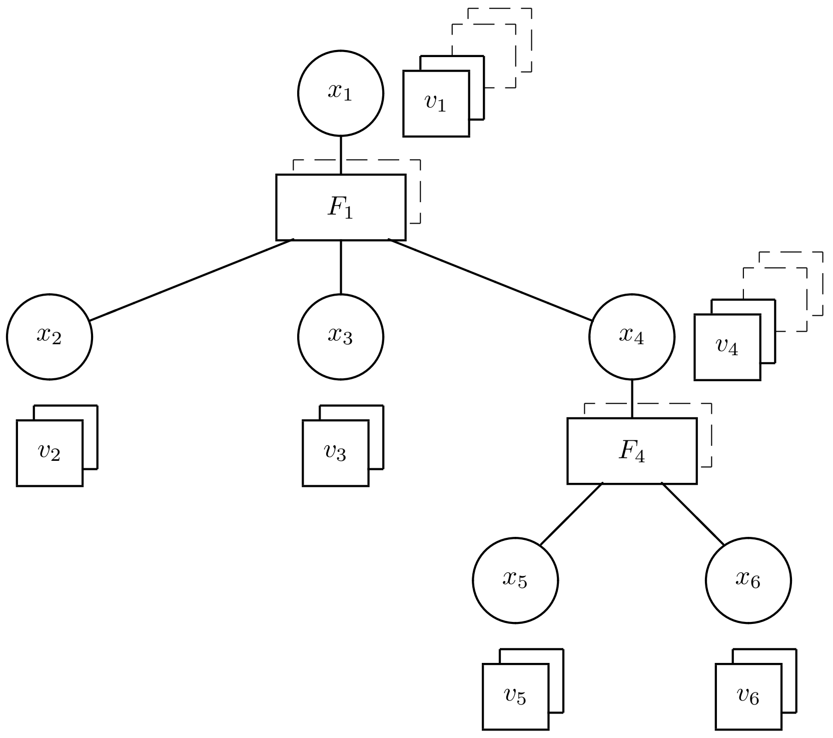

Figure 1: Criteria tree with utility functions and alternatives

Figure 1 illustrates a decision knowledge base after alternatives have been evaluated. There are two aggregate criteria, x_1 (the root) and x_4. The corresponding utility functions are F_1 and F_4. Note that in the presence of different decision making groups, more than one function can be defined for each aggregate criterion.

There are two alternatives shown in figure 1. They are described by values v_2, v_3, v_5 and v_6 that are assigned to basic criteria x_2, x_3, x_5 and x_6. Evaluation results v_4 and v_1 for each alternative have been obtained by functions F_4 and F_1, respectively. The evaluation is performed separately for each decision making group. Therefore, when there are more groups, a *set* of evaluation results, consisting of v_4 and v_1, is obtained for each group.

An example of a real-world criteria tree is shown in figure 2. The tree was designed in a project whose goal was to evaluate microcomputers with respect to their appropriateness for education in primary and secondary schools of the Republic of Slovenia (Rajkovič *et al.* 1985). Figure 2 actually presents only the subtree that was used to evaluate microcomputer *hardware*. The whole tree, which additionally included software and commercial criteria, was about twice as big. In figure 2, aggregate and basic attributes are represented by upper-case and lower-case letters, respectively, and followed by the corresponding value domains. According to the tree, microcomputer Hardware is determined by three main groups of criteria: Environment, Processor

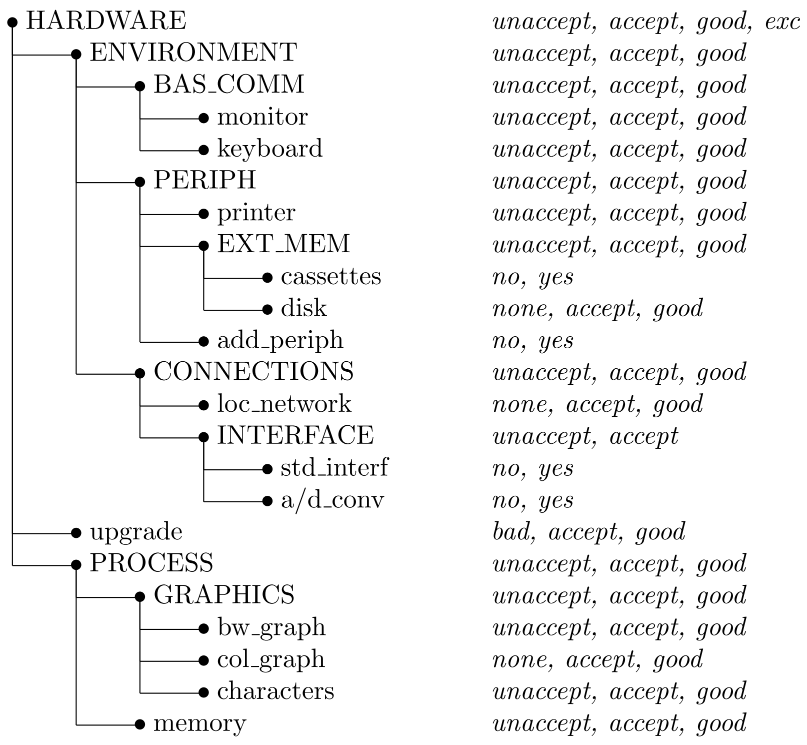
HARDWARE unaccept, accept, good, exc
ENVIRONMENT unaccept, accept, good
BAS_COMM unaccept, accept, good
monitor unaccept, accept, good
keyboard unaccept, accept, good
PERIPH unaccept, accept, good
printer unaccept, accept, good
EXT_MEM unaccept, accept, good
cassettes no, yes
disk none, accept, good
add_periph no, yes
CONNECTIONS unaccept, accept, good
loc_network none, accept, good
INTERFACE unaccept, accept
std_interf no, yes
a/d_conv no, yes
upgrade bad, accept, good
PROCESS unaccept, accept, good
GRAPHICS unaccept, accept, good
bw_graph unaccept, accept, good
col_graph none, accept, good
characters unaccept, accept, good
memory unaccept, accept, good

Figure 2: Criteria tree for the evaluation of microcomputer hardware

and capabilities to Upgrade the system. The first two are additionally decomposed. For example, the Environment is decomposed into Basic Communication, Peripherals and Connections.

MONITOR	KEYBOARD	BAS_COMM
unaccept	*unaccept*	*unaccept*
unaccept	*good*	*unaccept*
accept	*accept*	*accept*
accept	*good*	*accept*
good	*unaccept*	*unaccept*
good	*accept*	*good*

Table 1: Elementary decision rules for Basic Communications

Table 1 presents a simple set of elementary rules that define Basic Communications according to the quality of Monitor and Keyboard. Note the incompleteness of this table: some combinations of values of Monitor and Keyboard, for example Monitor=*good* and Keyboard=*good*, are undefined in the table.

The approach presented in this section has been implemented within an expert system shell DECMAK (Bohanec *et al.* 1983, 1987; Rajkovič *et al.* 1988) and thoroughly verified in practice (see sections 7 and 8). The shell interactively supports the decision maker in the knowledge acquisition and evaluation stages of the decision making process.

It is beyond the scope of this chapter to describe *knowledge acquisition* techniques that are used in DECMAK. They are based on an interactive dialogue for the acquisition of elementary decision rules. Additional tools are available the present the whole knowledge base to the decision maker using different viewpoints and levels of detail (Rajkovič *et al.* 1988, Bohanec and Rajkovič 1987, 1988).

The following sections are focussed on techniques and algorithms that are used in DECMAK to support the *evaluation stage* of the decision process. In this stage, the evaluation of alternatives, explanation of results and analysis of alternatives take place.

4 Evaluation of alternatives and explanation of results

In this stage of the decision making process, the knowledge base, which we assume it has been defined and verified in the previous stages, is used to evaluate alternatives. At the beginning, the decision maker describes each alternative $a \in \mathcal{A}$ by a vector of values that correspond to all basic attributes $x_{b_1}, x_{b_2}, \ldots, x_{b_m}$:

$$a = \langle v_{b_1}, v_{b_2}, \ldots, v_{b_m} \rangle, \quad v_{b_i} \in X_{b_i}, \quad i = 1, 2, \ldots, m.$$

In DECMAK, these descriptions are entered interactively, using a spreadsheet-like *editor of alternatives.*

It should be noted that DECMAK is also capable of handling incomplete and uncertain data about alternatives by means of probabilistic and fuzzy distributions of values (Bohanec *et al.* 1983). For simplicity, this feature is omitted from the following presentation.

After the alternatives have been described, they are *evaluated* by DECMAK. The evaluation proceeds from bottom to the top of the criteria tree. The value v_i of each aggregate criterion x_i is obtained by evaluating the corresponding utility function F_i. The function arguments are the values v_{i_j} that are assigned to all the descendants of x_i in the tree:

$$v_i = F_i(v_{i_1}, v_{i_2}, \ldots), \quad v_{i_j} \in X_{i_j}.$$

In order to evaluate F_i, DECMAK searches for the decision rule that exactly matches the arguments. If such a rule exists, it directly assigns the value to v_i. Otherwise, the regression analysis is applied over some of the remaining rules. Only the rules that slightly differ from the arguments of the function are considered. The Euclidean distance is used as a heuristic measure of difference between rules.

The evaluation of alternatives assigns values (utilities) to all the aggregate criteria in the tree. These values determine the adequateness of each alternative according to decision maker's preferences. The alternative that has got the most desirable values should be finally chosen.

However, some important questions usually emerge at this point:

- How were the results obtained? Why they are such? Are they appropriate?
- Are the results influenced by changes of basic criteria values and how? Which changes are required in order to get a better (or worse) alternative?
- What are the advantages and disadvantages of a particular alternative? How does it compare to some other alternative?

The evaluation results themselves are insufficient to answer these questions. Additional tools that help the user to *explain, justify and analyze* the decision are needed. As shown below, the rule-based approach offers quite useful support for such tasks.

Obviously, the evaluation process itself can be easily *explained* in terms of decision rules that have been triggered and criteria values that have been used. This is similar to the *How?* type of explanation in expert systems.

There is, however, a problem with the explanation of results obtained by the regression analysis. Due to its inherent "black-box" functioning, it is difficult to explain the computation itself. However, since it operates on a limited set of rules (usually, 2 to 5), these rules are shown to the user. According to our experience, the users are able to justify the appropriateness of the computation on the basis of this implicit information.

With DECMAK, the evaluation end explanation are performed interactively. The user can focus attention to particular aggregate criteria, inspect and/or change rules and follow the detailed explanation. This is often combined with *what-if* analysis, where the user changes descriptions of particular alternatives, reevaluates them and compares the results with the original ones.

5 Selective explanation and comparison of alternatives

The explanation described above answers the question: *How were the results obtained*? Its main role is to present a detailed overview of the evaluation process and to help the user in discovering errors in the knowledge base and descriptions of alternatives. However, there is a question that is usually more important in practice: *Why are the results such*? Unfortunately, the ultimate answer is hidden in rules; they are basic facts that can be meaningfully explained only by the person who defined them.

Nevertheless, some implicit and practically important answers can be gained by exploring the evaluation results and relations among them. For example, if an alternative got a bad final utility, one might be interested in sources (i.e., basic criteria values) that caused this result. For a better overview of an alternative it is also useful to find its most important advantages.

For this purpose, a *selective explanation* algorithm is used in DECMAK. Its main role is to explain a particular alternative in terms of its most advantageous or disadvantageous characteristics. The explanation is selective because, instead of all, usually quite numerous criteria, only the most relevant ones are shown in the form of subtrees of the whole criteria tree.

This algorithm operates by searching for such maximal connected subtrees of the whole criteria tree in which a particular alternative evaluates to *very bad* or, alternatively, *very good*. To determine whether a value is *very bad* or *very good*, a simple heuristics is used. Since the criteria domains are usually preferentially ordered (95% of domains used in about forty practical applications of DECMAK were such), the leftmost element of each domain is considered *very bad* and the rightmost *very good* by default. When this is inappropriate, the user is allowed to explicitly redefine the sets of *very good* and *very bad* values.

An example of selective explanation is shown in figure 3. The quality of hardware of a particular microcomputer is explained on the basis of the criteria tree from figure 2. The default settings have been used in order to determine *very bad* and *very good* values.

Almost the same algorithm can be used also for the *selective comparison* of alternatives, which is needed for:

- comparison of two different alternatives,
- discovering changes of one alternative during the *what-if* analysis, and
- comparison of the same alternative that was evaluated by utility functions of different decision making groups.

Advantages:

COL_GRAPH is *good*

INTERFACE is *accept*
    because STD_INTERF is *yes*
    because A/D_CONV is *yes*

ADD_PERIPH is *yes*

CASSETTES is *yes*

Disadvantages:

HARDWARE is *unaccept*
    because ENVIRONMENT is *unaccept*
        because BAS_COMM is *unaccept*
            because MONITOR is *unaccept*
        because CONNECTIONS is *unaccept*
            because LOC_NETWORK is *none*
    because PROCESS is *unacc*
        because GRAPHICS is *unaccept*
            because BW_GRAPH is *unaccept*
            because CHARACTERS is *bad*

Figure 3: Selective explanation of a microcomputer's hardware

The comparison algorithm differs from the explanation one only in the condition that must be satisfied in all nodes of the extracted subtrees. Here, the condition is satisfied if the first alternative is (strictly) better than the second one or, alternatively, the second one is better than the first. The relation *better* (represented by '>' in figure 4) is again determined from the preferentially ordered domains or explicitly given by the user. Figure 4 presents an example of selective comparison of two microcomputers.

Alternative A is better in:

BAS_COMM is better (*good* > *accept*)
    because MONITOR is BETTER (*good* > *accept*)

ADD_PERIPH is better (*yes* > *no*)

Alternative B is better in:

HARDWARE is better (*exc* > *accept*)
    because PROCESS is better (*good* > *accept*)
        because GRAPHICS is better (*good* > *accept*)
            because BW_GRAPH is better (*good* > *accept*)
            because COL_GRAPH is better (*good* > *none*)

PRINTER is better (*good* > *accept*)

Figure 4: Selective comparison of two microcomputers

6 Option generation

Another important aspect of the analysis and implementation of the decision is to find such small changes in descriptions of an alternative which cause degrading or upgrading its performance. The findings about changes that might degrade the alternative can be particularly useful in the stage of its implementation; once the most critical characteristics have been identified, it is much easier for the implementer to treat them carefully. On the other hand, the information about the changes that upgrade the alternative can serve a guideline for possible improvements of the alternative.

In DECMAK, such information can be gathered by an *option generator.* To describe the algorithm, let us concentrate on a single aggregate criterion y. Let $x_1, x_2, \ldots, x_k$ be its descendants in the criteria tree. Therefore, the relation between the criteria is that

$$y = F(x_1, x_2, \ldots, x_k).$$

We also assume that an alternative $a \in \mathcal{A}$ has been already evaluated on y and denote with

$$\mathbf{v} = \langle v_1, v_2, \ldots, v_k \rangle$$

the vector of values that have been assigned to $x_1, x_2, \ldots, x_k$ for this alternative. Let u be the utility of the alternative, obtained by

$$u = F(\mathbf{v}).$$

The task of the option generator is then to determine the smallest changes $\Delta\mathbf{v}$ of vector $\mathbf{v}$ which would lead to

$$F(\mathbf{v} + \Delta\mathbf{v}) = z,$$

such that $z > u$ for upgrading the alternative or $z < u$ otherwise.

At the beginning, the option generator implicitly completes the table of rules that define F by adding all the missing rules; the regression analysis from section 5 is used for this purpose. Then, the set of all rules that yield the value of z is determined. These are the rules of the form:

$$\textbf{if } x_1 = w_1 \textbf{ and } \ldots \textbf{ and } x_k = v_k \textbf{ then } y = z.$$

Each such rule corresponds to the assignment of values

$$z = F(w_1, \ldots, w_k) = F(\mathbf{w}).$$

Therefore, $F(\mathbf{w}) = F(\mathbf{v} + \Delta\mathbf{v})$, so

$$\Delta\mathbf{v} = \mathbf{w} - \mathbf{v}.$$

However, such $\Delta\mathbf{v}$ need not be the smallest one; it should be compared to all the difference vectors that are determined from the remaining rules which assign $y = z$. Therefore, all pairs of rules such that $z = F(\mathbf{w_1})$ and $z = F(\mathbf{w_2})$ are compared by the option generator. When upgrading alternatives, $\Delta\mathbf{v}_1$ that follows from the first rule is eliminated if $w_{1,i} \geq w_{2,i}$ for all $i = 1, 2, \ldots, k$. When finding the smallest changes that degrade alternatives, the above condition is replaced by $w_{1,i} \leq w_{2,i}$.

The extension of this algorithm from one aggregate criterion y to any subtree of y is straightforward: the algorithm should only be applied recursively on all the aggregate descendants of y.

An example of the option generator output is shown in figure 5. It was generated by applying the above algorithm on rules from table 1. Note, however, that this table is extremely simple, so the output is not as interesting and valuable as when treating more complex tables and subtrees.

```
Value 1: MONITOR=accept
Value 2: KEYBOARD=good
Utility:  BAS_COMM=accept

BAS_COMM becomes unaccept
          if ( MONITOR becomes unaccept ) or
          if ( MONITOR becomes good and
               KEYBOARD becomes unaccept ).

BAS_COMM becomes good
          if ( MONITOR becomes good and
               KEYBOARD becomes accept ).
```

Figure 5: Output of the option generator

7 Applications

The DECMAK system has been applied in about 40 practical decision making problems. They varied from simple, personal decisions, like job or car selection, to complex group decision problems, such as:

1. evaluating mainframe computer systems (8 applications so far);
2. selection of computers for schools (4 applications);
3. microcomputer selection;
4. evaluation of production control software;
5. data base management system selection (2 applications);
6. trading partner selection;
7. evaluation of applications for nursery schools;
8. matching people to jobs;
9. expert team selection;
10. project feasibility estimation;
11. performance evaluation of public enterprises.

The first part of the list (applications 1 to 5) indicates that DECMAK has been extensively used in decisions related to computers. The main reason for this is the computer science background of the authors of the system. In these problems, it was convenient for the authors to play a dual role of decision analysts and computer consultants. Nevertheless, the applications numbered 6 to 11 show that the usefulness of DECMAK is not limited only to computer-oriented problems.

The complexity of a particular decision problem is reflected in the size of the knowledge base and the number of considered alternatives. These data for some of the above problems are presented in table 2.

Application	Number of		
	Basic criteria	Aggregate criteria	Alternatives
1.a	20	9	6
b	67	45	2
c	17	9	3
d	35	24	8
2.a	11	6	6
b	12	9	10
c	36	19	24
5.	74	29	3
6.	22	12	>20
7.	8	5	>2000
9.	16	8	164
10.	34	19	1
11.	12	9	55

Table 2: The complexity of some DECMAK applications

8 Discussion

One of the main motivations for combining multi-attribute decision making with the knowledge-based approach in DECMAK was to improve the quality of decision making by the aid of *explanation.* There are extremely rare decision situations where the criteria and alternatives are so clear that they directly lead to optimal solutions without any further analysis and explanation. Usually, a great deal of uncertainty and missing information is involved, important criteria may be overlooked or treated inappropriately, etc. Therefore, the obtained results should not be accepted until they are carefully investigated, justified and understood by the decision maker.

The proposed approach is based on a structured and explicitly articulated knowledge base which can be argued, verified and documented. Its structure is specially adapted for multi-attribute decision problems. Although the structure is relatively simple compared to common expert systems, it has been found quite general and flexible in practice. In the stage of *knowledge acquisition*, the main advantages are (1) easy acquisition of elementary decision rules by an interactive man-machine dialogue and (2) powerful capabilities for explaining the whole knowledge base by means of machine learning techniques. These are presented and discussed elsewhere (Bohanec and Rajkovič 1988; Rajkovič and Bohanec 1991).

In the *evaluation stage* of the decision making process, the knowledge base is applied consistently on all alternatives in order to obtain and explain their utilities. The evaluation approximately corresponds to reasoning in common expert systems. However, due to the specifics of the problem, it is much more "function-oriented"; it basically calculates function values. There is no inference based on logic, theorem proving, etc. The order of evaluation of rules is strictly determined by the structure of criteria.

In all practical applications of DECMAK (section 7), some kind of explanation was needed (and done) in the evaluation stage. Usually, at least some of the evaluation results were in disagreement with the decision maker's expectations. It was necessary to investigate them in more detail and find out the reasons for the disagreement, which could have been caused by erroneous rules, inappropriate descriptions of alternatives, overlooked criteria or wrong expectations. In this respect, the explanation by tracing the triggered rules (section 4) has been found useful, but not as much as we had originally expected. It was invoked quite rarely, mainly to explain unexpected values assigned to a single aggregate criterion. In our opinion, the main reason for this is in low and nonselective informativity of this kind of explanation. The decision maker is usually at the same time the author and user of the rules. For this reason, the triggered rules hardly provide any new information for him.

In order to discuss alternatives within the decision making group and to justify the decision, an overall (aggregate) explanation of each alternative is needed. In practice, it seems that the *selective explanation* algorithm (section 5) suits extremely well for this purpose. It was used in all practical decision problems. The decision makers considered it an important support tool for the explanation and verification of the decision. Its results were also very useful in preparing reports and other presentations related to the decision. Usually, only the translation of the printouts (such as the ones in figure 3) into ordinary sentences was sufficient to provide a comprehensible explanation.

It seems that the main power of the selective explanation is in its ability to extract and group together the criteria and values that most relevantly determine the quality of a particular alternative. However, in the current implementation of DECMAK, the "relevance" of criteria is determined by a very simple heuristic inspection of criteria domains. Further research is suggested at this point to find more general and robust measures of relevance, for example by taking into account also the underlying utility functions.

The *option generator* (section 6) is usually applied after a limited set of possible candidates for the best alternative has been identified. These are then thoroughly analyzed, mainly to isolate critical characteristics that can easily degrade them in the implementation stage. When the decision makers are in a position to influence the alternatives, for example to require some modifications of computer hardware from the distributor, the possibilities of upgrading them are also interesting for investigation. According to our experience, the usefulness of the option generator heavily depends on the decision problem. There were problems where it was not needed at all, for example in selecting an expert team and evaluating application for nursery schools. Its role was more important in decisions related to computers and public enterprises. However, there was a problem where a decision had to be made whether to continue or abandon a large software development project. All aspects of this project were thoroughly analyzed almost exclusively by the option generator.

9 Conclusion

The decision making approach presented in this chapter combines two techniques, multi-attribute decision methods and expert systems. It is based on an explicit articulation of knowledge about a particular decision problem. The knowledge is represented by tree-structured criteria and utility functions, defined by elementary decision rules.

The approach is supported by an expert system shell DECMAK. It consists of tools which support interactive knowledge acquisition and evaluation of alternatives. The main emphasis is on the explanation and analysis of the evaluation process.

DECMAK has been verified in about forty practical decision making situations. The results confirmed the importance of the explanation facilities. Their availability motivated the decision makers to think about, verify, justify and explore decisions. We strongly believe this is one of the key elements that can improve the effectiveness and justification of decisions.

Among the algorithms presented in this chapter, the selective explanation one has been found the most valuable in practice. Its main advantage is the ability to explain alternatives by selecting only the most relevant information and presenting it to the user in a structured form.

References

Alter, S.L., 1980. *Decision support systems: Current practice and continuing challenges.* Reading: Addison-Wesley.

Bohanec, M., Bratko, I. and Rajkovič, V., 1983. An expert system for decision making. In: H.G. Sol (ed.), *Processes and tools for decision support.* Amsterdam: North-Holland.

Bohanec, M. and Rajkovič, V., 1987. An expert system approach to multi-attribute decision making. In: M.H. Hamza (ed.), Proceedings of IASTED conference on expert systems. Anaheim: Acta Press.

Bohanec, M. and Rajkovič, V., 1988. Knowledge acquisition and explanation for multi-attribute decision making, Proceedings of the eighth international workshop Expert systems and their applications, Vol. 1, Avignon.

Chankong, V. and Haimes, Y.Y., 1983. *Multiobjective decision making: theory and methodology.* Amsterdam: North-Holland.

Efstathiou, J. and Mamdani, E.H., 1986. Expert systems and how they are applied to industrial decision making. In: Mitra, G. (ed.), *Computer assisted decision making.* Amsterdam: Elsevier.

Fox, M.S., 1985. Knowledge representation for decision support. In: Methlie, L.B. and Sprague, R.H. (eds.), *Knowledge representation for decision support systems.* Amsterdam: North-Holland.

French, S., 1986. *Decision theory: an introduction to the mathematics of rationality.* New York: Wiley.

Humphreys, C.P. and Wisudha, D.A., 1987. *Methods and tools for structuring and analysing decision problems.* Technical Report 87-1. London: The London School of Economics and Political Sciences.

Keen, P.G.W. and Scott Morton, M.S., 1978. *Decision support systems: an organizational perspective.* Reading: Addison-Wesley.

Keeney, L.R. and Raiffa, H., 1976. *Decisions with multiple objectives: preferences and value tradeoffs.* New York: Wiley.

Klein, M. and Methlie, L.B., 1990. *Expert systems: a decision support approach.* Reading: Addison-Wesley.

Lévine, P., Pomerol, M.-J. and Saneh, R., 1990. Rules integrate data in a multicriteria decision support system. *IEEE Transactions on Systems, Man, and Cybernetics* **20**, 678–686.

Phillips, L.D., 1986. Decision analysis and its applications in industry. In: Mitra, G. (ed.), *Computer assisted decision making.* Amsterdam: Elsevier.

Rajkovič, V. *et al.* , 1985. Evaluation of microcomputers for primary and secondary schools in Slovenia (in Slovene language). Ljubljana: Zavod SR Slovenije za šolstvo.

Rajkovič, V., Bohanec, M. and Batagelj, V., 1988. Knowledge engineering techniques for utility identification. *Acta Psychologica* **68**, 271–286.

Rajkovič, V. and Bohanec, M., 1991 Decision support by knowledge explanation. In: Sol, H.G. and Vecsenyi, J. (eds.), *Environments for supporting decision processes.* Amsterdam: North-Holland.

Turban, E., 1988. *Decision support and expert systems.* New York: Macmillian.

Vari, A. and Vecsenyi, J., 1988. Concepts and tools of artificial intelligence for human decision making. *Acta Psychologica* **68**, 217–236.