KNOWLEDGE ACQUISITION AND EXPLANATION FOR MULTI-ATTRIBUTE DECISION MAKING *

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

[(1)] 'Jožef Stefan' Institute, Jamova 39, 61000 Ljubljana, Yugoslavia,
tel. +38 61 214 399, tlx. 31296 yu jostin
[(2)] School of Organizational Sciences, Prešernova 11, 64000 Kranj, Yugoslavia

Abstract: A decision making approach which combines multi-attribute decision making techniques with expert systems is described in the paper. 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. In the paper, the discussion is mainly concentrated on the part of the shell which deals with interactive knowledge acquisition and knowledge explanation. The corresponding algorithms are given and discussed.

Key-words: expert systems, multi-attribute decision making, knowledge representation, knowledge acquisition, knowledge explanation.

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(s),
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. They range from everyday personal decisions to more complex problems like selection of the most appropriate technology, project management and planning, matching people to jobs and many others. The complexity of such problems usually originates in

*A reformatted version of the paper published in the *Proceedings of the 8th International Workshop 'Expert Systems and Their Applications AVIGNON 88', Vol. 1, 59–78, Avignon, 1988.*

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

Many methods and computer programs have been developed in order to help decision makers solve more or less complex problems (Humphreys and Wisudha 1987). They are usually 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 may also be considered systems that solve and explain complex repetitive decisions (Efstathiou and Mamdani 1986).

The main role of decision methods and tools is to *support* decision makers in

- organizing and systematizing the facts, data and knowledge that influence the decision,
- consistently applying these upon all alternatives, and in
- further analysis and optimization of the alternatives.

In this paper we show how multi-attribute decision making methods can be *combined* with expert system technology in order to obtain a better quality in terms of *decision knowledge acquisition and explanation*, where an expert system plays the role of a *cognitive support tool* for decision making. In the next section, the concept of multi-attribute decision making is presented. Section 3 gives a formal description of knowledge representation that is used in the proposed approach. The methods for knowledge acquisition, knowledge explanation and evaluation of alternatives are presented in sections 4, 5 and 6, respectively. After a brief description of practical applications of this approach (section 7), methods for knowledge acquisition and explanation are further discussed in section 8.

2 MULTI-ATTRIBUTE DECISION MAKING

In *multi-attribute* decision making, the decision problem is decomposed into a number of smaller, less complex subproblems (Keeney and Raiffa 1976, Chankong and Haimes 1983, French 1986). Alternatives are decomposed onto different dimensions, usually called *attributes, criteria, goals,* etc. These are evaluated independently. The total utility of an alternative is finally obtained by some aggregation procedure. Alternatives are ranked according to utility values, where a higher value means a better alternative.

In multi-attribute terms, the decision problem is therefore described by:

1. a set of *alternatives* $\mathcal{A} = \{A_1, A_2, \ldots\}$;
2. a set of *criteria* $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$; each criterion $X_i \in \mathcal{X}$ is further described by its *name*, x_i, and the *domain* D_i of values it may hold;
3. *partial utility functions* $f_i : D_i \rightarrow R_i,\ i = 1, 2, \ldots, n$;
4. a *global utility (aggregation) function* $F : R_1 \times R_2 \times \ldots \times R_n \rightarrow R$.

In order to evaluate alternatives, each alternative $A \in \mathcal{A}$ is first described by a vector of values

$$\mathbf{v} = [v_1, \ldots, v_n],\ v_i \in D_i,\ i = 1, 2, \ldots, n.$$

The partial utility functions are then used to determine the decision maker's *preference* P_i of the alternative according to each particular criterion X_i:

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

The preference P_i is usually expressed as a number within a certain interval R_i, which is often the same for all the criteria, for example [0,1].

Finally, the total utility of the alternative is calculated:

$$F(A) = F(P_1, P_2, \ldots, P_n).$$

In practice, the most commonly used formula for F is a weighted sum where weights w_i are given by the decision maker:

$$F(P_1, P_2, \ldots, P_n) = \sum_{i=1}^{n} w_i P_i.$$

The above description gives only the main ideas of multi-attribute decision making which may be treated differently in different methods. For example, criteria may be further structured into trees. There are also large differences among the methods in the representation and assessment of alternatives and utility functions (Humphreys and Wisudha 1987).

However, the most common drawback of existing multi-attribute methods is in the need of translating the decision makers' *knowledge* about a particular decision problem into (analytical) functions and numbers. The translation itself may be a problem. Furthermore, such models are poorly suited for

- verification, justification and explanation of the model itself, and
- explanation of the obtained evaluation results.

Explanation facilities are vital in decision making practice, especially when no a priori optimal solution exists and when more than one person is involved in decision making. Explanation is practically impossible with analytical utility functions because of difficulties with intuitive interpretation (understanding) of them.

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 the 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 formulae.

3 KNOWLEDGE REPRESENTATION

Knowledge representation for multi-attribute decision making is based on a *tree of criteria* (also called *semantic tree*). Such a tree describes the structure of a particular decision problem and serves as a 'skeleton' for utility functions and evaluation of alternatives. With increasing depth of the tree, more specific decision subproblems are considered. Let us define:

Criteria tree $\mathcal{T}$ is a pair $(\mathcal{X}, S)$ where:

X is a set of *criteria* $\{X_1, X_2, \ldots, X_n\}$; each criterion $X_i \in \mathcal{X}$ is further described by its *name*, x_i, and the *domain* D_i of values it may hold;

S is a mapping $\mathcal{X} \rightarrow 2^{\mathcal{X}}$ ($2^{\mathcal{X}}$ is the power set of $\mathcal{X}$). For each criterion, the mapping S defines a set of its immediate descendants (i.e. 'sons') in the tree. It must satisfy the following requirements:

$$\bigcup_{i=1}^{n} S(X_i) = \mathcal{X} - X_1$$

and

$$X_k \notin S^*(X_k),\ k = 1, 2, \ldots, n.$$

S^* denotes the transitive closure of S. $S^*(X)$ therefore represents the set of *all* descendants of X.

The above requirements ensure that $\mathcal{T}$ is a connected and directed graph without cycles. In practice, it is usually further restricted to a tree, although this is unnecessary from the formal point of view. According to the requirements, X_1 is the *root* of $\mathcal{T}$. The criteria X for which $S(X) = \emptyset$, are *leaves* of the tree and are also called *basic criteria*. All other criteria are *aggregate criteria*.

The domains $D_1, D_2, \ldots, D_n$ of the corresponding criteria $X_1, X_2, \ldots, X_n$ are assumed finite and discrete. They generally consist of words, not numbers. For example, words such as 'good' or 'high' may be used in the domains. For numerical quantities, intervals of values should be used as (discrete) domain

elements. It is also recommended (but not required) to order the domains by preference from 'bad' to 'good' values, for example bad, acceptable, good, excellent or, for price, high, medium, low.

Utility functions are the second component of the decision knowledge base. Their purpose is to define the influence of lower-level criteria on the higher-level ones. For each aggregate criterion $X_k \in \mathcal{X}$, the corresponding utility function

$$F_k : D_{i1} \times D_{i2} \times \ldots \times D_{im} \rightarrow D_k$$

should be specified by the decision maker. $D_{i1}, \ldots, D_{im}$ are the domains of all sons of X_k, i.e. $\{X_{i1}, X_{i2}, \ldots, X_{im}\} = S(X_k)$.

It often happens that different decision groups with different interests are involved in the decision making process. For this purpose, each group is allowed to define its own utility functions. Therefore, as shown in figure 1, more than one function may be defined for a particular criterion. However, the same criteria tree must be agreed and used by all groups. Roughly speaking, the groups are still required to "speak the same language" in order to be able to compare and adjust their standpoints.

In the proposed approach, utility functions are specified by the decision maker in the form of *elementary decision rules*. Let $X_{i1}, X_{i2}, \ldots, X_{im}$ be the sons of an aggregate criterion X_k. Then, the function

$$X_k = F_k(X_{i1}, \ldots, X_{im})$$

is defined by a set of rules of the form

if $X_{i1} = v_{i1}$ **and** $\ldots$ **and** $X_{im} = v_{im}$ **then** $X_k = v$,

where v_{ij} and v denote single values taken from the corresponding criteria domains.

The rules must not conflict with each other, i.e. only one rule with a given conditional part may be defined. On the other hand, it is not necessary to define all possible rules, i.e. the rules that cover all combinations of values in their conditions. The rules that define a particular function are usually represented in a tabular form (see, for example, table 1).

After the criteria tree and utility functions have been defined, evaluation of alternatives may begin. 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 of all alternatives. These values, especially the ones that have been assigned to the root of the criteria tree, are finally used as a guideline for selection of the best alternative.

Figure 1 illustrates a decision knowledge base after alternatives have been evaluated. The criteria X_1 to X_6 are structured into a tree where X_1 is the root and X_4 is the second (lower-level) aggregate criterion. The utility functions F_1 and F_4 correspond to these aggregate criteria. If there are more decision making groups, more than one function may be defined at each aggregate node. Alternatives are described by values of basic criteria, i.e. X_2, X_3, X_5 and X_6.

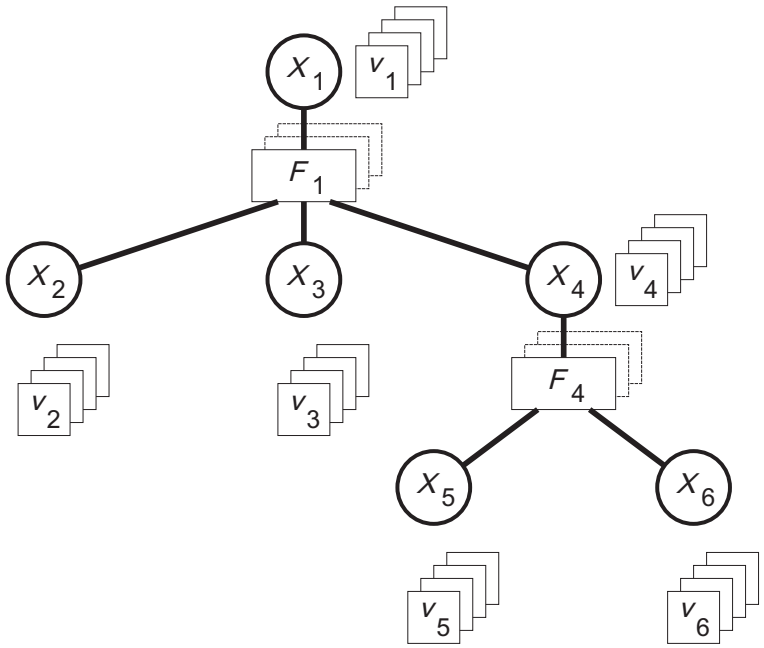

Figure 1: Criteria tree, utility functions and alternatives

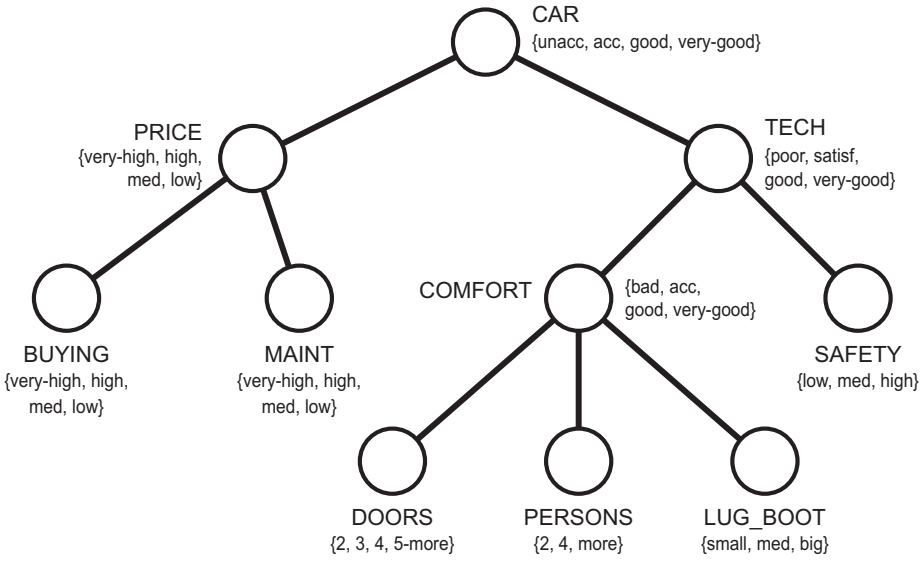

Figure 2: Criteria tree for the car selection problem

COMFORT	SAFETY	TECH
bad	low	poor
bad	med	poor
bad	high	poor
acc	low	poor
acc	high	good
good	low	poor
good	med	good
good	high	v-good
v-good	low	poor
v-good	med	good

Table 1: Elementary decision rules for Technical characteristics

In figure 1, four alternatives are assumed. Evaluation results v_4 and v_1 for each alternative have been obtained using functions F_4 and F_1, respectively. If there are more decision making groups, the alternatives are evaluated separately for each group, using the corresponding functions. For each group, different sets of evaluation results, v_4 and v_1, are obtained in this case (which is not explicitly shown in figure 1).

A simple (and even simplified for the sake of demonstration) example of the criteria tree for car selection is shown in figure 2. According to this tree, the quality of cars is measured by two main groups of criteria: price and technical characteristics. The price is determined by buying and maintenance price. Technical characteristics are decomposed into safety and comfort, which further depends on number of doors, size of a car (measured as number of persons that fit in the car) and size of the luggage boot. In figure 2, the names and domains of these criteria are presented. The domains are given in braces. An example of rules that specify technical characteristics of a car according to its comfort and safety is presented in table 1. Note that not all the possible combinations of values of comfort and safety have been defined by the decision maker in this case. Tables such as table 1 should be specified for all aggregate criteria, in this case also for PRICE, COMFORT and CAR.

The approach presented in this section has been implemented within an expert system shell named DECMAK (Bohanec et al. 1983, 1987, Rajkovič et al. 1986) and thoroughly tested in about thirty practical decision situations (see sections 7 and 8). The main role of the shell is to actively support the decision maker in knowledge acquisition and evaluation stages of the decision making process. In the following sections, the implemented methods for knowledge acquisition, verification and explanation are presented.

4 KNOWLEDGE ACQUISITION

According to the components of the decision knowledge base (figure 1), there are two stages of the knowledge acquisition process:

- criteria tree design, and

- acquisition of utility functions.

The DECMAK shell offers quite a limited support of the first stage. The reason is because criteria trees tend to be extremely problem-dependent. Automatic or semi-automatic construction of them requires a *deep understanding* of the decision problem that we are not able to handle by the computer yet. This problem is evident in all multi-attribute methods which start by criteria tree design. It may also be compared to the field of machine learning (Michie and Bratko 1986) where the domain description in terms of attributes must be first provided by the expert. Automatic machine learning may take place only after this has been done. By now, the highest level of computer support at this stage is limited to an *editor* of criteria trees.

Acquisition of *utility functions* is much more actively supported by the system. The functions are acquired by means of elementary decision rules (see section 3). An *editor* of tables (such as table 1) can be used for this purpose. However, it can be a tedious process to fill-in large tables. In order to make this process more effective, a question-answer dialogue has been designed. It consists of two algorithms named ASK and ANSWER.

Before we describe these algorithms, let us define some notational conventions that will be used in this and the following section. As each utility function is assigned to a particular aggregate criterion, the discussion can be without the loss of generality restricted to only one aggregate criterion, $Y \in \mathcal{X}$. Let its sons in the criteria tree be $X_1, X_2, \ldots, X_k$, and $X_1, X_2, \ldots, X_k = S(Y)$. We shall also denote:

F - utility function $Y = F(X_1, X_2, \ldots, X_k)$
D_Y - domain of Y
D_i - domain of X_i for $i = 1, 2, \ldots, k$
$y \in D_Y$
$x_i \in D_i$

As domains D_Y and D_i are discrete and somehow (usually preferentially) ordered by the decision maker during the time of their creation, we shall assume them enumerated, therefore

$y \in 1, 2, \ldots, m,\ m = |D_Y|$, and
$x_i \in 1, 2, \ldots, m_i,\ m_i = |D_i|,\ i = 1, 2, \ldots, k.$

A particular decision rule will be denoted by

$(x_1, x_2, \ldots, x_k) \Rightarrow y$

where the left hand side will be called a *condition* and y a *value* of the rule.

Let us now describe the *ASK* algorithm. Its purpose is to generate conditional parts of rules and present them as questions to the decision maker who only answers by giving a value. During practical experiments with ASK it turned out that the order of posing the questions was important. Randomly stated questions, i.e. subsequent questions with large differences in their conditions, confused the user and slowed down the acquisition process. In order to stay as long as possible in a given context and to allow the user to compare the current

question with previously given answers, only small variations in subsequent conditions are allowed. This may be also seen from the research in psychology, where human ability to concentrate in the depths and unability to jump from one subject to another were shown (Lindsay and Norman 1977).

Up to now, the best results were obtained by the following algorithm:

choose initial condition I;
while asking not stopped by the decision maker **and**
table of rules is not full **do**
generate question Q by small (i.e. first +1, then -1)
variations of all values of I;
if there is no such question Q **then** {make a large step}:
find condition C such that:
- in the table there is no rule of the form $C \Rightarrow c$
and
- C maximizes the Euclidean distance between C and the condition of the closest already defined rule;
$I := C$; $Q := C$;
end if;
ask the question Q;
if Q was answered by y
then enter the rule $Q \Rightarrow y$ into the table
else the user need not answer the question;
end if;
end while.

The initial condition I is chosen by

if the table of rules is empty
then $I := (1, 1, \ldots, 1)$
else $I := R$, where $R \Rightarrow x$ is the last rule entered into the table;
end if.

From the algorithm it may be seen that the decision maker need not answer all the questions (some of them may be irrelevant in a given situation). He or she may stop the acquisition process at any time and/or select the new context of questions by explicitly entering a complete rule into the table. If the change of the context must be made automatically, the largest possible step is done in order to explore all parts of the decision space.

In order to reduce errors during acquisition of rules, this process is continuously monitored by a *monotonicity checker*. Criteria domains are usually ordered by preference, and thus the majority of utility functions (in practice, about 90%) increase or remain stable when their arguments increase. When a newly entered rule violates monotonicity, the user is warned, but no specific action is undertaken to prevent entering it into the table.

ANSWER is the second important algorithm used in rule acquisition. It is in a sense 'symmetric' to ASK. Given a conditional part of a rule and other already

specified rules, it calculates a value of the rule. Afterwards, this rule may but need not be entered into the table. The calculated value may be overridden by the user as well. This algorithm is important in *verifying* the knowledge base. Once appropriate behaviour has been observed by means of ANSWER, rule acquisition process may be stopped and the corresponding table of rules need not be completely filled-in. ANSWER may be invoked automatically also in the stage of *evaluation* of alternatives (section 6).

Several different methods were practically tested within the ANSWER calculation procedure. They mainly based on regression (linear, quadratic, cubic) or on machine learning algorithms that build decision trees, namely ID3 (Quinlan 1979) and CART (Breiman et al. 1984). It was estimated (on a quite subjective basis, we must admit) that least 'obvious mistakes' were made by a simple method based on linear regression. This method considers rules from a particular table as points in a multidimensional space and approximates them by a (hyper)plane. In this approximation, only rules that nearly surround (in Euclidean terms) the rule for which the value is to be determined, are taken into account. The algorithm to compute the unknown value y of rule $R \Rightarrow y$ is therefore:

$S := \emptyset$;
repeat
    add to S all $P \in T - S$ such that $|R - P| = \min_{Q \in T-S} |R - Q|$;
    compute y by linear regression using rules from S;
until y computed **or** $S = T$.

In this algorithm, T denotes the set of all defined rules (i.e. the table). S is the set of rules that nearly surround rule R. $|R-P|$ denotes the Euclidean distance between the rules R and P.

The disadvantage of using regression in the above algorithm is its 'black-box' behaviour which can not be easily explained to the decision maker. However, after the computation has been done, the set S contains the selected rules that were used in the computation. As S is usually much smaller than the complete table T, the rules from S may be listed and relatively easily reviewed by the decision maker.

5 KNOWLEDGE EXPLANATION

After utility functions have been acquired, they can be immediately used for evaluation and analysis of alternatives (section 6). However, it turned out to be a good practice to review and justify the knowledge base before proceeding to the next stage. This is important not only to find erroneous rules, but also to further rethink the whole knowledge base in order not to overlook some important aspects of the decision making problem and to ensure that the knowledge base really expresses the decision maker's preference knowledge. The decision maker must be able to *understand* the *whole* knowledge base and to *explain* it to himself, to other members of the decision making group(s) and even to

```
SAFETY
 low: TECH = poor
 med:
   COMFORT
     bad: TECH = poor
     acc: ?
     >=good: TECH = good
  high:
    COMFORT
     bad: TECH = poor
     acc: TECH = good
     good: TECH = v-good
     v-good: ?
```

Table 2: Decision tree for Technical characteristics

people not directly involved in the decision making process. Unfortunately, this important stage is almost completely omitted from 'traditional' multi-attribute techniques.

In DECMAK, several algorithms are used for the purpose of knowledge explanation. Their main role is to *translate* elementary decision rules into different representations which show *the same* knowledge from different viewpoints and at different levels of detail.

Elementary rules themselves are the most detailed representation of the knowledge. However, large tables of these simple rules tend to become difficult to handle, interpret and verify. In DECMAK, they are made more compact and/or easily understandable by the following three groups of algorithms:

- inductive learning algorithms,
- graphics, and
- linear regression analysis.

Inductive learning algorithms (Michie and Bratko 1986) treat elementary decision rules as *examples* of particular alternatives or their parts with the known values. On the basis of these examples, they produce more compact and/or generalized description of the underlying knowledge. Usually, decision trees or more complex rules are generated (here, the term decision tree refers to decision trees as they are considered in inductive learning (Quinlan 1979), not in decision analysis (French 1986)).

In order to make a *decision tree* out of elementary rules, Quinlan's ID3 algorithm (Quinlan 1979) is used in DECMAK. Table 2 shows an example of a decision tree that was generated by ID3 using elementary rules (examples) from table 1. It may be interpreted as follows: If safety of a car is low, then technical characteristics are poor. If safety is medium, comfort of a car must be further considered. If it is bad, technical characteristics are poor as well. If comfort

is good or better, technical characteristics are good. However, if comfort is acceptable, there is a question mark in the tree which indicates that such information has not been specified by the decision maker and that ID3 was unable to generalize the knowledge at this point. Such information may be valuable to discover loosely defined parts of the decision space, but may also be avoided by combining ID3 with the regression method described in section 4.

The advantage of using decision trees is their ability to structure the decision space. Points (rules) in decision space that are close to each other are also closely structured in the tree. They can be easily compared.

However, it turned out in our practical experiments that some users disliked decision trees, mainly because they were so different from the original tables. This problem can be quite easily overridden by translating a decision tree into a table. Each path from the root of the tree to one of its leaves may be represented by a row in this table. Unfortunately, such tables tend to be non-optimal, i.e. there exist tables with less rows. For this reason, another method for generating *aggregate tables* is applied in DECMAK. It is actually an adapted (and quite simple) version of AQ inductive learning algorithms (Michalski and Larson 1983). This method directly translates elementary decision rules into more aggregate ones. The only difference between elementary and aggregate rules is that within the latter, appearance of *intervals* of values in their conditions is allowed, i.e.

if $X_1 \in I_1$ **and** $\ldots$ **and** $X_k \in I_k$ **then** $Y = v$,

where I_i denotes an interval of values of the corresponding domain D_i.

Aggregate rules are iteratively generated for each $y \in D_Y$. The algorithm tries to find the lower number of aggregate rules that replace all elementary rules with value y. In the case that more equivalent solutions emerge, a heuristic criterion of 'understandability' is applied to choose one. Aggregate rules are considered 'understandable' if intervals I_i in their conditions equal to the corresponding criteria domains or cover at least one marginal (very good or very bad) value. Small internal intervals are not preferred.

The algorithm is as follows:

for each $y \in D_Y$ **do**
    while there exist uncovered elementary rules $P \Rightarrow y$ **do**
        choose uncovered elementary rule $P \Rightarrow y$;
        generate all aggregate rules that
            - cover the chosen rule, **and**
            - cover some other elementary rules with value y, **and**
            - do not include undefined parts of the decision space;
        choose such aggregate rule which maximizes the number of
            covered elementary rules;
        if there are more such aggregate rules **then**
            among these choose the most 'understandable' one;
        write the chosen aggregate rule;
    end while
end for.

COMFORT	SAFETY	TECH
-	low	poor
bad	-	poor
acc	high	good
>=good	med	good
good	high	v-good

Table 3: Aggregate rules for Technical characteristics

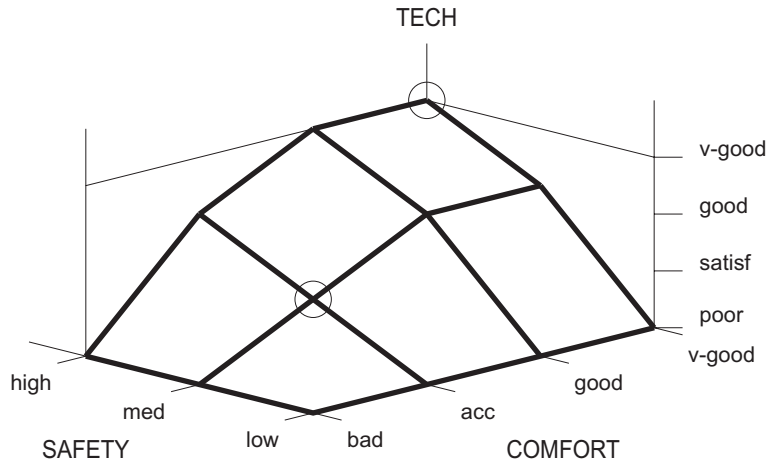

Figure 3: Graphic representation of rules from table 1

Aggregate rules which were generated out of elementary rules in table 1, are shown in table 3. Note that '-' indicates the whole range of values, i.e. represents any value of a particular criterion. '>=good' means 'good or better' or, equivalently, 'at least good'.

Let us briefly mention the remaining two tools used in knowledge explanation, *graphic display* of utility functions and *regression analysis*. The main goal of graphics is to present the overall shape of a particular function. For example, figure 3 presents the same function as given in tables 1 to 3. Note circles around two points on the surface. They mark values which were not explicitly given by the decision maker but estimated by the ANSWER algorithm (section 4).

Regression analysis is applied in order to get a high-level description of utility functions, i.e. to represent them without any details. All points of a particular function are approximated by a (hyper)plane. Gradient of this plane, normalized to the unit space, is then transformed into *weights* (relative importances) of particular criteria. Weights are usually expressed as percentages. For example, the weights of criteria COMFORT and SAFETY are, according to the function in figure 3, 45.9% and 54.1%, respectively.

6 EVALUATION AND ANALYSIS OF ALTERNATIVES

In this stage of the decision making process, the knowledge base is actually used to solve the decision problem. Alternatives should be measured and described by values of basic criteria. The alternatives are then evaluated according to the defined utility functions. Generally, the alternative that got the most desirable evaluation results, should be considered the best.

In DECMAK, the evaluation process proceeds from bottom to the top of the criteria tree. Evaluation of a particular function starts with searching for the appropriate rule. If one is not found, the ANSWER algorithm is applied to compute the function value.

Evaluation of alternatives is the main, but not the only task of a decision making system that should be performed at this stage. Evaluation results should be at least justified and explained. In DECMAK, as it is common in expert systems, these activities are emphasized and actively supported. DECMAK can

- explain the evaluation process in terms of rules that were triggered,
- explain each alternative by its advantages and disadvantages,
- compare two alternatives by selecting only those criteria which significantly influence the difference, and
- perform 'what-if' and sensitivity analysis.

DECMAK can also deal with incomplete and uncertain data about alternatives by means of probabilistic and fuzzy distributions of values (Bohanec et al. 1983).

It is beyond the scope of this paper to further describe numerous techniques which are implemented in DECMAK for these purposes. They are treated in more detail in (Rajkovič et al. 1986, Bohanec and Rajkovič 1987).

7 APPLICATIONS

Up to now, DECMAK has been applied in about 30 practical decision making situations. The problems varied from simple, personal decisions (like job, car or photo camera selection) to complex group decision problems usually connected with large investments:

1. evaluation of computer systems for an enterprise (5 applications);
2. selection of computer hardware and/or software for schools (4 applications);
3. microcomputer selection;
4. evaluation of production control software;

PROB-LEM	BASIC CRITERIA	AGGREGATE CRITERIA	ALTER-NATIVES
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 4: Illustration of complexity of some practical decision problems

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. evaluation of a project feasibility;
11. performance evaluation of enterprises.

As it is evident from this list, DECMAK has been intensively used in decisions connected with computers. The main reason for this is the background of the authors of this system which is in computer science. It was convenient for them to play a dual role in decision making processes, being decision analysts and computer consultants at the same time. However, applications numbered 6 to 11 clearly show that usefulness of this approach is not limited to computer-oriented problems.

The complexity of a particular decision problem is usually expressed by the number of criteria and the number of alternatives. These data for some of the above problems are summarized in table 4.

8 DISCUSSION

It is difficult to objectively measure the quality of a given method or approach in decision making. After 'the best' alternative has been chosen and implemented in practice, it becomes extremely difficult to compare this alternative with alternatives that were rejected by the decision. These alternatives are 'dead' and

there is no information on how they would have performed in practice under the same circumstances. It is therefore difficult to determine whether the decision was really 'the right' one or not.

Let us therefore discuss some 'indirect' impacts of the proposed decision making approach to the quality of decision making. As it is common to all decision making techniques, it helps in organizing in systematizing the decision maker(s) knowledge about a particular decision making problem. Here, the main goal is not to overlook criteria that significantly influence the decision and to apply the constructed model consistently upon all alternatives. Effective use of the method and its ability to help understand, justify and explain the decision are also important.

Time that was required to make the decisions mentioned in section 7 varied quite a lot according to the complexity of the problem, familiarity of the decision makers with the problem (i.e. amount of available knowledge), availability of data about alternatives, motivation of the decision makers, documentation requirements, etc. For simple personal problems, at most few hours were spent. For more complex problems, this varied from 2 (applications 1b and 10) to 20 (application 1d) days. Most commonly, the decision making process took 3 to 6 days with the following time schedule:

- 1 to 2 days: criteria tree design,
- 1 to 2 days: definition of utility functions,
- 1 to 2 days: evaluation and analysis of alternatives.

This is more than two days that are usual for decision making conferences where 'traditional' decision analysis is used (Phillips 1986). There are two reasons for this. First, more time is needed to define decision rules than, for example, just fill-in weights into a predefined formula. But in return we gain a flexible support in justifying and explaining the knowledge itself and in explaining the decision. However, exploring these facilities in detail also increases the overall decision making time.

When analyzing separate stages of the decision making process, the first one, *design of the criteria tree*, appears the most critical. It directly influences the relevance and success of the final decision. Selecting relevant criteria and structuring them into tree is a difficult creative task which requires a deep understanding of the problem at hand. It is still more art than science. Errors in tree design may but need not be uncovered in the subsequent stages of the process. Further research is needed in order to develop tools that would actively and on a higher semantic level support this design. Some solutions might be acquired by intelligent, problem-oriented interfaces and/or predefined knowledge bases to be only customized for a particular decision problem.

The *acquisition* of utility functions with the support of DECMAK tools is much less problematic. Elementary decision rules, although simple, have been found very flexible and appropriate for acquisition of preference knowledge. This may be justified by findings in the field of machine learning (Michie and Bratko 1986). Each elementary rule may be also viewed as an *example* of a (real or fictional)

alternative. By defining the value of a rule, the decision maker actually evaluates this alternative. However, such evaluation is not done on the whole range of criteria, but only on those which are determined by the position of the utility function within the criteria tree. It has been shown in machine learning that this way of knowledge acquisition resolves the Feigenbaum's bottleneck problem of knowledge articulation (Feigenbaum 1981, Michie and Bratko 1986).

Knowledge acquisition by examples may still require significant amount of time. Tools like ASK (section 4) become extremely important in this regard. Moreover, we believe that without such a tool the proposed approach would not have been feasible. In practice, the use of ASK helped us significantly reduce the time required to define a utility function (i.e. table such as table 1 or larger). Now, this can be done in the range of minutes, typically 1 to 20 depending on the complexity of the domain. However, in complex criteria trees, utility functions are numerous and the whole process can still last a day or two. It also appeared that work with ASK was quite demanding for humans. It was difficult for them to concentrate on questions longer than, say, two hours at once.

Yet another limitation of human information processing emerged at this stage. If the number of functions' arguments (i.e. descendants in the tree) exceeded four, it became very difficult to define appropriate rules. Knowledge acquisition time increased and errors were introduced. This should be taken into account during the design of criteria trees in order not to group more than four criteria together.

ASK's counterpart, ANSWER, is not so vital in knowledge acquisition. Its main purpose is to make use of rule tables that are only partially filled-in. In DECMAK, it is based on regression which gives relatively good results, but offers quite limited possibilities for explanation. Usually, 40% to 60% of all possible rules in a table should be defined in order to obtain the desired answers. Further research is suggested at this point in order to get a method that would work correctly with less rules and that its actions would be more easily explained.

The group of tools for *knowledge explanation* (section 5) is again very important in practice. In (but not limited to) decision making, knowledge must be first very well understood by the decision maker himself. On this basis, human learning and knowledge refinement cycles may take place (Michie and Bratko 1986). The knowledge must also be communicated among the people directly or indirectly involved in creation and/or use of the knowledge base. For these purposes, different viewpoints (representations) of the same knowledge at different levels of detail are required.

DECMAK offers quite a wide range of knowledge explanation tools. According to the opinion of users of the system, the most useful of them are algorithms that generate aggregate rules (table 3) and graphics (figure 3).

It is again difficult to measure the impact of knowledge explanation tools to the quality of final decisions. On the basis of numerous refinements and improvements of the knowledge bases that resulted directly from different explanation views and that would otherwise had been overlooked, we strongly believe that these tools make decisions better. At least, they are better justified and explained.

9 CONCLUSION

The decision making approach presented in this paper combines two technologies, multi-attribute decision methods and expert systems. The result of the decision making process is structured and explicitly articulated knowledge base, which can be argued, verified and documented. Knowledge base structure is specially adapted for multi-attribute decision problems. It consists of tree-structured criteria and utility functions, represented by elementary decision rules. This relatively simple structure has been found sufficiently general and flexible in about thirty practical decision situations.

The approach is supported by an expert system shell named DECMAK. It consists of tools which actively support knowledge acquisition and its use for evaluation of alternatives. The main emphasis is on explanation of evaluation results and explanation of the knowledge itself. For the latter, tools that translate the knowledge base into different representations at different level of detail are implemented.

The proposed approach must not be considered a replacement of all 'traditional' multi-attribute methods neither of expert systems in decision making. According to our practical experience, it is best suited for complex decision situations, where

- large number of criteria is involved (say, more than 15),
- judgmental (qualitative) decision making is necessary,
- knowledge explanation and explanation of the results are needed (e.g. in group decision making),
- there are at least 2 to 3 days available for the decision, and
- there is an IBM PC/XT/AT or VAX computer available (without computer tools like ASK or knowledge explanation algorithms, this approach is practically unfeasible).

For example, expert systems with more specific knowledge-base structure and inference engine might be preferred in complex and repetitive decision making problems. On the other hand, 'traditional' multi-attribute techniques are probably better for personal decision making, for problems with prevailing numeric criteria and well defined objectives, and for decisions which are to be made extremely quickly.

10 REFERENCES

Alter, S.L.: DECISION SUPPORT SYSTEMS - CURRENT PRACTICE AND CONTINUING CHALLENGES, Addison-Wesley, 1980.

Bohanec, M., Bratko, I., Rajkovič, V.: AN EXPERT SYSTEM FOR DECISION MAKING, In Sol, H.G. (ed.): Processes and Tools for Decision Support, North-Holland, 1983.

Bohanec, M., Rajkovič, V.: AN EXPERT SYSTEM APPROACH TO MULTI-ATTRIBUTE DECISION MAKING, IASTED International Conference on Expert Systems, Geneva, 1987.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: CLASSIFICATION AND REGRESSION TREES, Wadsworth International Group, Belmont, California, 1984.

Chankong, V., Haimes, Y.Y.: MULTIOBJECTIVE DECISION MAKING: THEORY AND METHODOLOGY, North-Holland, 1983.

Efstathiou, J., Mamdani, E.H.: EXPERT SYSTEMS AND HOW THEY ARE APPLIED TO INDUSTRIAL DECISION MAKING, In Mitra, G. (ed.): Computer Assisted Decision Making, Elsevier Science Publishers B.V. (North-Holland), 1986.

Feigenbaum, E.A.: EXPERT SYSTEMS IN THE 1980S, In Bond, A. (ed.): State of the Art Report on Machine Intelligence, Pergamon-Infotech, 1981.

French, S.: DECISION THEORY, Ellis Horwood Limited, 1986.

Humphreys, C.P., Wisudha, D.A.: METHODS AND TOOLS FOR STRUCTURING AND ANALYSING DECISION PROBLEMS, Decision Analysis Unit, The London School of Economics and Political Sciences, Tech. Rep. 87-1, London, 1987.

Keen, P.G.W., Scott Morton, M.S.: DECISION SUPPORT SYSTEMS - AN ORGANIZATIONAL PERSPECTIVE, Addison-Wesley, 1978.

Keeney, R.L., Raiffa, H.: DECISIONS WITH MULTIPLE OBJECTIVES, John Wiley & Sons, New York, 1976.

Lindsay, H.P., Norman, A.D.: HUMAN INFORMATION PROCESSING: AN INTRODUCTION TO PSYCHOLOGY, Academic Press, 1977.

Michalski, R.S., Larson, J.: INCREMENAL GENERATION OF VL_1 HYPOTHESES: THE UNDERLYING METHODOLOGY AND THE DESCRIPTION OF PROGRAM AQ11, Report ISG 83-5, University of Illinois at Urbana-Champaign, 1983.

Michie, D., Bratko, I.: EXPERT SYSTEMS: AUTOMATING KNOWLEDGE ACQUISITION, Addison-Wesley, 1986.

Phillips, L.D.: DECISION ANALYSIS AND ITS APPLICATIONS IN INDUSTRY, In Mitra, G. (ed.): Computer Assisted Decision Making, Elsevier Science Publishers B.V. (North-Holland), 1986.

Quinlan, J.R.: DISCOVERING RULES BY INDUCTION FROM LARGE COLLECTIONS OF EXAMPLES, In Michie, D. (ed.): Expert Systems in the Microelectronic Age, Edinburgh University Press, 1979.

Rajkovič, V., Bohanec, M., Efstathiou, J.: RANKING MULTIPLE OPTIONS WITH DECMAK, UNICOM Seminar on Effective Decision Support: Resolving Problems with Multiple and Conflicting Objectives, London, 1986.