Contributions to Methodology and Statistics
A. Ferligoj and A. Kramberger (Editors)
Metodološki zvezki, 10, Ljubljana: FDV 1995

Comparison of Different Decision Making Approaches

Andrej Mrvar[1]

Abstract

Multicriteria decision making approaches help us to solve complex problems – searching for the best solution according to several criteria. There are different ways in which algorithms transform preference relation into utility function and then aggregate criteria to compound criterion. According to that the multicriteria decision making approaches can be divided into two groups:

- algorithms based on weighted sum,
- knowledge–based system algorithms (KBS).

An example of weighted sum algorithm (Saaty method) and an example of knowledge–based system (Ripple down rules) are described and compared in the paper.

1 Introduction

Decision making is a process of selecting the best solution from the set of all possible solutions. This problem is trivial if it considers only one criterion. But most of the real problems treat several criteria. This means that we try to find the best solution from the set of all possible solutions according to all criteria which we are interested in. Two main problems can be identified in multicriteria decision making:

- transformation of preference relation into utility function,
- aggregation of criteria.

Algorithms for decision making can be divided into two groups according to these two problems:

- algorithms based on weighted sum,
- knowledge–based system algorithms (KBS).

In this paper both types of algorithms are discussed and compared.

[1]Faculty of Social Sciences, University of Ljubljana, P.O. Box 47, 61 109 Ljubljana, Slovenia

Table 1: Matrix of pairwise comparisons

	price	cons.	speed	safety	comfort
price	1	4	6	3	7
cons.	1/4	1	4	2	3
speed	1/6	1/4	1	1/2	2
safety	1/3	1/2	2	1	3
comfort	1/7	1/3	1/2	1/3	1
Weights	0.51	0.21	0.08	0.14	0.06

2 Algorithms based on weighted sum

Probably the best known method for aggregation of criteria in multicriteria decision making is based on weighted sum. The importance of criteria is expressed in the form of weights – real numbers between 0 and 1. Higher weight means that a criterion is more important.

When we use this approach the main problem is how to find appropriate weights for considered criteria. Normally the people are not able to divide 100% into smaller parts that would add up to 100 and where each part would represent the importance of a criterion.

One approach that solves this problem is the method which was developed by T. L. Saaty (1988) and which is called the Saaty method. This approach enables us to get weights of importance of criteria and weights of priorities of solutions according to one criterion in an automatic way. How? ... All pairs of criteria have to be compared – that is $n(n-1)/2$ comparisons, where n denotes the number of criteria. Scale 1 to 9 is used, where 1 means that two criteria are equally important, 2 means that the first criterion is a little more important, and so on, 9 means that the first criterion is absolutely more important than the other. If the second criterion is more important than the first one, inverse numbers (1/2, ..., 1/9) are used. In this way we get the square matrix of pairwise comparisons. The weights of all criteria are obtained by solving the 'eigenvalue problem' for this matrix. The weights are actually components of the eigenvector that corresponds to the largest eigenvalue. This eigenvalue is always real, single and positive.

2.1 An example

Let us, for example, solve the problem of selection of the most desirable car (Mrvar, 1992) with respect to: price, fuel consumption, speed, safety and comfort. Let us suppose that all pairwise comparisons were measured. They are given in Table 1.

We can easily find some properties of the matrix of pairwise comparisons:

- Diagonal values are all 1, because diagonal values are comparisons of a criterion to itself.

- Symmetrical elements are the reciprocal numbers (as much as the first criterion is more/less important than the second, the second is less/more important than the first).
- We have given the value 4 to comparison price/consumption which means that the price is more important than consumption.
- The price row has all the values larger than 1. This means that the price is the most important criterion.

The components of eigenvector give us the importance of each criterion. From Table 1 we can see that the price is really the most important criterion, the second most important criterion is consumption, while safety, speed and comfort are less important.

2.2 Comparison of Saaty method with other weighted sum methods

There are at least two reasons why the Saaty method is more suitable method than the other weighted sum methods:

- Normally, in the real problems, the importance cannot be expressed in the form of percentages. E. g., in the problem of selecting the most appropriate job, it is difficult to determine the priority vector for criteria like the enjoyment of work, salary, distance from home, etc. Usually people have problems to estimate the priorities of these criteria directly in percentages, because this is not a natural way of human thinking. It is easier to compare different criteria than to measure the importance of each criterion. The Saaty method provides exactly what is needed in this situation – it provides the systematic approach which enables us to get weights directly from pairwise comparisons.
- The second advantage of this method is that it also provides us a measure of consistency of the estimation of pairwise comparisons. Actually $n - 1$ comparisons are enough to get the entire $n \times n$ matrix of pairwise comparisons. We can get the other values in the matrix by using the simple computation, because there is a kind of transitivity between judgments. The redundant comparisons are used as a measure of consistency. The algorithm can decide whether the judgments are consistent enough to continue the decision process. If the judgments are inconsistent the algorithm will compute which judgment is the most inconsistent and will suggest the user a better value for it. Therefore, the Saaty method can also be used as a test of how well the user actually knows the problem which he wants to solve.

3 Knowledge–based system algorithms

The algorithms based on weighted sum select the best solution from the set of all possible solutions. The algorithms in the knowledge–based system approach are very different from those based on weighted sum. In this approach the machine learning program which generates the rules for classification into classes is used. It is based on a given set of learning examples. The obtained results can be expressed in many different ways:

- decision trees,
- production rules,
- Ripple down rules (RDR).

Decision trees and production rules are well known methods for knowledge representation. Therefore the Ripple down rules are presented in greater detail.

RDR were introduced by P. Compton (Compton and Jansen, 1988). This method is a combination of decision trees and production rules. Both methods are combined to get the most suitable, compact and understandable knowledge representation. It combines the structure of decision trees and conjunctive normal form (CNF) of tests on criteria which is used in production rules' presentation. (CNF is a logical statement where the tests on criteria are connected by the logical operator 'AND'.) Each RDR classifies an object into one of few exclusive classes, but it also allows some exceptions to the rule. When it finds out that an object satisfies all properties to be classified in a certain class, the process of classification is not finished, because it is also possible that the object has some additional properties which define an exception to the rule.

3.1 RDR algorithm

Let us suppose that a set of learning examples for which the values of all criteria and the resulting class are known. First Quinlan's ID3 algorithm (Quinlan, 1986) is used to generate decision tree. The main idea of ID3 algorithm is the following one: The algorithm tries to find the decision tree that will explain the learning examples and will allow the classes of unseen examples to be predicted. In each step of building the tree it selects the most informative criterion – that is, the criterion which will divide the set of learning examples into subsets so that the most similar objects will be classified into the same class. The algorithm selects the criterion which decreases impurity (objects that are not accurately classified) in the subset the most. The theory of information (entropy) is used to determine which criterion is the most important. The algorithm is repeated until all examples in any subset belong to the same class.

Then the algorithm Ghidora (Catlett, 1992) for transformation of decision tree into Ripple down rules is used. The main idea of this algorithm is the following: Ghidora algorithm starts with an initial RDR set consisting of just the default rule specifying the majority class. It chooses the next rule by examining all the nodes of the tree where the majority class deviates from the class given by the RDR set. Then it selects the node which, if a rule were added to cover it, would give the greatest reduction in errors on the learning set.

3.2 An example

RDR are applied to the following problem: Determine the appropriateness of programmers according to their personal characteristics (PERS) (Rajkovič and Bohanec, 1991). The following criteria and their possible values are considered:

- selfinitiativeness (INI) – insufficient, normal, strong.
- creativity (CREA) – no creativity, normal creativity, ability to develop new algorithms, ability to develop new techniques.
- communication with users (COMM) – bad, normal, good, very good.

These criteria are used for classifying each programmer into one of the following exclusive classes: inappropriate, appropriate, very appropriate programmer.

The input for this example consists of 44 learning examples (see Table 2). E. g., let us look at the learning example 15 in Table 2. It tells us that if the programmer's initiativeness is strong and his communication is normal, but if he has no creativity, this programmer is still inappropriate.
17 Ripple down rules obtained as the result of the conversion algorithm are presented in Table 3. The first rule which is added to the RDR set is the default rule. The frequencies of personal characteristics in the set of learning examples show that the most (23) programmers are inappropriate. Therefore the first statement is that all programmers are inappropriate. But if we look at the examples more precisely we will see that this is not always the case. There are 21 programmers that are at least appropriate. Therefore we try to find the CNF of tests on criteria that will define an exception to the first rule and will decrease the error (21) the most. It is possible to find, that if the creativity is normal and the communication is very good, the programmer will be appropriate and the error is decreased by 2 (there are 2 such programmers). But this rule has also one exception: if the initiativeness is strong too, the programmer will not only be appropriate but very appropriate. The algorithm continues till the error is 0 (all learning examples are covered).

3.3 Comparison of RDR method with other Knowledge–based systems

The comparison of Ripple down rules with production rules or decision trees shows that RDR method has many advantages:

First of all the number of RDR is much smaller than the number of production rules or the number of nodes in decision tree. Because of this RDR are more suitable for treating larger problems, where decision trees and production rules are very difficult to interpret.

The examples on which the program was tested are presented in Table 4. The first example has already been discussed. The program was also tested on medical domains which are often treated by machine learning artificial intelligence programs. The number of nodes in ID3 tree is smaller than the number of learning examples

Table 2: Learning examples

No.	INI	CREAT	COMM	PERS
1.	insuf	no	bad	inapprop
2.	normal	no	bad	inapprop
3.	strong	no	bad	inapprop
4.	insuf	normal	bad	inapprop
5.	normal	normal	bad	inapprop
6.	strong	normal	bad	inapprop
7.	insuf	newalgo	bad	inapprop
8.	normal	newalgo	bad	inapprop
9.	strong	newalgo	bad	inapprop
10.	insuf	newtech	bad	inapprop
11.	normal	newtech	bad	inapprop
12.	strong	newtech	bad	inapprop
13.	insuf	no	normal	inapprop
14.	normal	no	normal	inapprop
15.	strong	no	normal	inapprop
16.	insuf	normal	normal	inapprop
17.	insuf	no	good	inapprop
18.	normal	no	good	inapprop
19.	strong	no	good	inapprop
20.	insuf	normal	good	inapprop
21.	insuf	no	very good	inapprop
22.	normal	no	very good	inapprop
23.	strong	no	very good	inapprop
24.	normal	normal	normal	approp
25.	strong	normal	normal	approp
26.	insuf	newalgo	normal	approp
27.	normal	newalgo	normal	approp
28.	insuf	newtech	normal	approp
29.	normal	newtech	normal	approp
30.	normal	normal	good	approp
31.	strong	normal	good	approp
32.	insuf	newalgo	good	approp
33.	normal	newalgo	good	approp
34.	insuf	newtech	good	approp
35.	insuf	normal	very good	approp
36.	normal	normal	very good	approp
37.	insuf	newalgo	very good	approp
38.	insuf	newtech	very good	approp
39.	strong	newalgo	normal	very approp
40.	strong	newtech	normal	very approp
41.	normal	newtech	good	very approp
42.	strong	normal	very good	very approp
43.	normal	newalgo	very good	very approp
44.	strong	newalgo	very good	very approp

Table 3: Ripple down dules from domain: PERS

```
0  21   1  inapprop
2  17   3    approp if (CREAT=newalgo)
3  14   4      inapprop if (COMM=bad)
1   5  12      very approp if (COMM=very good)
1   4  13        approp if (INI=insuf)
1   6  11      very approp if (COMM=normal) and (INI=strong)
1  11   6    approp if (CREAT=normal) and (COMM=normal)
1  10   7      inapprop if (INI=insuf)
1   9   8    approp if (CREAT=normal) and (COMM=good)
1   8   9      inapprop if (INI=insuf)
2  19   2    approp if (CREAT=normal) and (COMM=very good)
1   7  10      very approp if (INI=strong)
2  12   5    approp if (CREAT=newtech) and (COMM=normal)
1   3  14      very approp if (INI=strong)
1   2  15    approp if (CREAT=newtech) and (COMM=good)
1   1  16      very approp if (INI=normal)
1   0  17    approp if (CREAT=newtech) and (COMM=very good)
```

in all cases but the reduction is much higher if RDR is used.
Catlett (1992) found some other advantages of RDR:

- Experts can easily choose the most important set of rules. For every RDR obtained by the conversion algorithm it is possible to compute the so called 'importance ranking' which estimates the expected benefit of the rule to the accuracy of the set of rules.

- The context of any set of rules is available nearby. In the case of production rules one should examine all rules of higher order to find which steps have led to the decision (classification of an object), while in RDR one should look just at all rules to which the rule is an exception.

Table 4: Testing domains

Domain	No.learn. examples	No.nodes ID3 tree	No.leaves ID3 tree	No.RDR
Personal char.	44	38	27	17
Primary tumour	237	153	84	60
Lympography	104	77	60	17
Hepatitis	108	62	49	19
Breast cancer	202	135	85	20

- It is always possible to use just as many rules as it is important for the classification accuracy. Importance ranking and reduction of error tell us which rules can be left out of our examination and how the error will be increased. This method is called pruning of decision tree and is often used in artificial intelligence. This is especially important in the case of large decision trees where it is better to use a little less accurate, smaller tree than the entire tree which is difficult to interpret. Bottom criteria in the tree usually do not improve the accuracy very much.

4 Conclusion

There is no general rule of thumb which decision approach to use. We can conclude this comparison of decision making approaches with some general suggestions for the users:

- Weighted sum algorithms are very appropriate for decision making when the problems are not complex. But even if the problems are complex the Saaty method is an appropriate method if we include hierarchical approach (Saaty, 1988; Mrvar, 1992). In this case similar criteria should be combined into one compound criterion.

- Weighted sum algorithms are not suitable if the considered criteria are not monotonous. This means that there are some situations when an extreme value of one criterion is enough to make a decision.

Two general rules can help in choosing the most appropriate approach:

- Choose the decision making approaches that best fit your problem.

- Among them, choose the approach that you understand the best (there is no sense in using an approach of which you heard about that it is the best, but you do not understand it enough).

References

[1] Catlett, J. (1992): Ripple-Down-Rules as a Mediating Representation in Interactive Induction. In *Proceedings: 2nd Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop: JKAW'92*, Japanese Society for AI, Kobe and Hotoyama, 155-170.

[2] Compton, P. and Jansen, R. (1988): Knowledge in context: a strategy for expert system maintenance. In *Proceedings AI'88: 2nd Australian Joint Artificial Intelligence Conference*, Adelaide, Australia: Springer-Verlag, 292-306.

[3] Mrvar, A. (1992): *Saatyjev večkriterijski odločitveni postopek*. Ljubljana: Faculty of Electrical Engineering and Computer Science.

[4] Quinlan, J. R. (1986): Induction of decision trees. *Machine Learning*, **1**, 81-106.

[5] Rajkovič, V. and Bohanec, M. (1991): Decision Support by Knowledge Explanation. In H. G. Sol and J. Vecsenyi (Eds.): *Environments for Supporting Decision Processes.* Amsterdam: North-Holland, 47-57.

[6] Saaty, L.T. (1988): *Multicriteria Decision Making - The Analytic Hierarchy Process.* Pittsburgh: RWS Publications.