Revision of Qualitative Multi-Attribute Decision Models

Martin Žnidaršič*
Marko Bohanec**

Jožef Stefan Institute
Department of Intelligent Systems
Ljubljana, Slovenia
*Email: martin.znidarsic@ijs.si
**Email: marko.bohanec@ijs.si

Abstract

Data-driven revision of decision models is defined as follows: given an existing model and a set of data items, revise the model to match the data items. In this paper, we propose and experimentally evaluate a method for the revision of qualitative hierarchical multi-attribute models in DEX methodology. The revision method is automatic, but limited to the modification of utility functions. Using a simple experimental model and simulated data, we show that the method is valid and that it improves the classification accuracy of the models.

Keywords

Decision support, qualitative multi-attribute decision models, data-driven model revision

1. INTRODUCTION

Decision models are an essential part of decision analysis. Usually a lot of effort is dedicated to the construction of a suitable and useful model. Expert knowledge and data that describes the decision problem or known solutions are carefully combined into a model.

The use of the model depends on the characteristics of decision problem. Some models are used only once, when a difficult decision has to be thoroughly analysed (BRS 1997, Greenberg et al. 2002). On the other hand, there are models built for continuous use on a regular, e.g. daily, basis (Zupan et al. 2001, Michalowski et al. 2003). Most of the models that are used continuously have to be regularly revised to reflect the new state of decision problem as well as possible. Building a model is a demanding, time consuming and expensive process. Revising an existing one is not much easier. Although the actual changes in the model are usually minor, this process requires: gathering new data, the evaluation of changes and their effect, reimplementation of decision support tools and verification of the new behaviour.

Models for continuous use are usually data-dependent and their updating can be automated to some extent, if contemporary data about decision problem is available. Nowadays the abundance of available fresh information and the advances in data analysis methods enable us to use data in various ways and to automate even as creative processes as learning, pattern recognition, customer support and many others (Han & Kamber 2001).

For qualitative multi-attribute models, there exist methods for their data-based construction from scratch (Zupan et al. 1999). However, these methods are complex, computationally demanding, they require a large quantity of data and often require an active involvement of an expert. The task addressed in this paper is simpler: given an already developed model and a few data items, revise the model so as to match the new data as closely as possible. We are not aware of any other existing methods of data-driven revision of qualitative decision models.

We tested the idea of automatic revision of decision models on hierarchic qualitative models of DEX methodology (Bohanec & Rajkovič 1990). An automatic revision procedure for these models was designed, implemented and tested. A simple model was created for a decision problem to represent the original model. New data was simulated by data from a slightly altered original model. The automatic revision procedure was then applied to the original model with new data and the revised model was evaluated. Experimental results proved that the proposed revision method works and adequately adapts the original model to new data.

DEX methodology is briefly presented in section 2. In section 3, the problem of automatic revision is defined and explained. The proposed revision method for qualitative models is presented in section 4. In section 5, the experimental setting and results are described. Conclusions are given in section 6.

2. DEX METHODOLOGY

DEX is one of multi-attribute decision modelling methodologies (Saaty 1980, Bohanec & Rajkovič 1990, Clemen 1996, Triantaphyllou 2000, Turban & Aronson 2001). Multi-attribute decision models (MADM) are used to evaluate, compare and study alternatives. Examples of alternatives are for instance cars, job candidates, office locations, etc. Usually we try to select the most appropriate alternative for our goals, the one with the highest utility. Decision problem-solving with MADM is based on a hierarchical decomposition of the problem. Alternatives are hierarchically decomposed into subconcepts (or aggregate attributes) and finally to a finite set of basic attributes. Utilities of aggregate attributes are evaluated with functions, which depend on the corresponding attributes located on the lower levels of the hierarchy. The MADM tools usually allow the analysis (alternative ranking, sensitivity analysis, what-if analysis) and graphical representation of the decision problem.

DEX was developed at Jožef Stefan Institute (Bohanec & Rajkovič 1990). Unlike traditional methodologies (Saaty 1980), DEX uses qualitative variables instead of numerical, what makes it suitable for less formalized decision problems. Utility functions in DEX are adjusted to qualitative variables and therefore represented with if-then rules, which are usually given in tabular form (Figure 2). This qualitative approach proved to be very useful in practice, since DEX was used in many real-world decision problems (Bohanec & Rajkovič 1999, Bohanec, Rajkovič & Cestnik 2003).

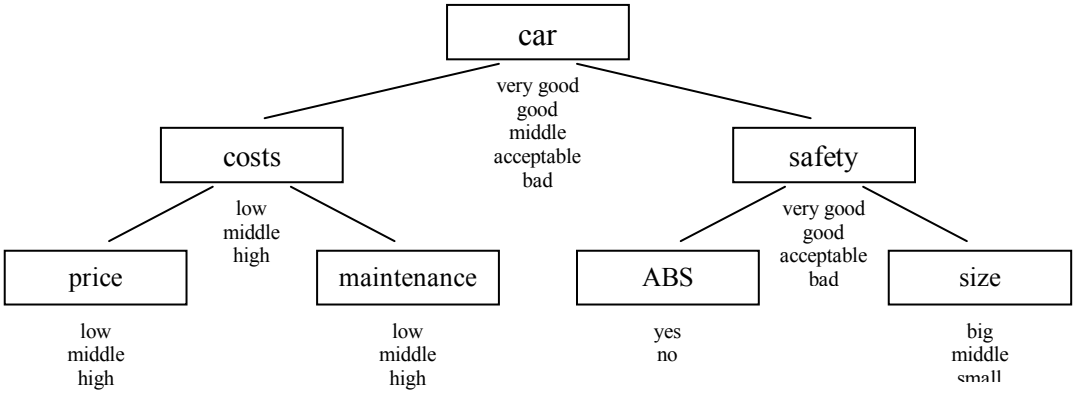

Figure 1: Hierarchy of attributes in a DEX decision model for car purchase.

In Figure 1, the hierarchy of attributes is shown for a simple decision setting we will use in this paper. It represents a decision problem of a selection of the most appropriate car to buy. The evaluation of the goal attribute "car" is decomposed into individual evaluation of two subconcepts we are interested in, "costs" and "safety". Each of them is further decomposed into basic attributes. Values of the attributes are written below each attribute in Figure 1. The aggregate attributes values are calculated as values of utility functions that have lower-level attribute values as an input. The tabular representation of the utility functions for our example case are shown in Figure 2. Notice that the values of basic attributes are not evaluated with utility functions, but they represent the basic features of cars and must be provided by the user.

price	*maintenance*	***costs***
low	low	low
low	middle	low
low	high	middle
middle	low	low
middle	middle	middle
middle	high	high
high	low	middle
high	middle	high
high	High	high

ABS	*size*	***safety***
no	small	bad
no	middle	acceptable
no	big	good
yes	small	bad
yes	middle	good
yes	big	very good

costs	*safety*	***car***
low	very good	very good
low	good	good
low	acceptable	middle
low	bad	bad
middle	very good	good
middle	good	good
middle	acceptable	acceptable
middle	bad	bad
high	very good	good
high	good	middle
high	acceptable	bad
high	bad	bad

Figure 2: Utility functions for our example case.

Attributes in DEX are usually, but not necessarily, ordered. The ordering implies that the corresponding utility functions are *monotone*: when the values of an ordered attribute increase, the utility function increases, too, or remains constant.

3. AUTOMATIC REVISION

The construction of decision support models is usually done by decision analysis experts and decision domain experts. It is a demanding, expensive and time-consuming process. The construction of decision models in general can not be fully automated, since it involves subjective, political, creative and similar elements. However, attempts were made to automate the construction of models or parts of models that can be sufficiently described with data (Zupan et al. 1999).

Decision support models that are used many times in similar but somewhat different situations (usually at different times) must be occasionally revised to guarantee an up-to-date reflection of decision problem environment. The revision of decision models is usually nearly as demanding as their construction. Each change of the model must be verified whether it is admissible and whether its effect improves the models to fit the decision environment.

We believe that this process could be automated in some situations, particularly when the decision model is frequently used and there is data available that describes some actual decisions. The use of decision models in such decision problem settings is currently rare, but it might become more frequent as the amount of data recorded in complex decision making problems increases. In some domains this is happening in present time (for instance real-estate evaluation, portfolio selection, etc.), but the use and revision of decision support models is still rare.

Besides a general acknowledgement of the usefulness of decision tools, the procedures which at least partly automate some of the processes could help to increase the number of decision support users. We propose an automatic revision of decision models as a valuable approach to adjusting the decision models in some decision support situations. This approach facilitates the revision of decision models, but demands useful data to be available and does not guarantee the optimal solution. The method for data-driven revison that we developed and tested, works only with hierarchical qualitative decision models.

The scope of revison on such models can be either wide or narrow, depending on how thorough revision changes are desirable. We can define the following types of revision, regarding the scope of changes to the model:

- revision of structure, values and functions;
- revision of values and functions;
- revision of functions.

The revision processes that are higly flexible (have a wide scope of changes) are more difficult to automate. The hierarchical structure of the model is its basic characteristic and we usually do not want to change it. Structural changes are very difficult to perform and can alter the model in many unwanted ways. Although the ability of performing considerable changes in the model is sometimes useful, thorough changes indicate that a major change occurred in decision problem environment. In such situations, a more elaborate revision with expert involvement may be more suitable than an automated approach.

Revision methods that change aggregate attributes values and utility functions are less flexible, but are more feasible for automation and have a scope of changes that is wide enough for successful adjustment of the models to new circumstances. The revision method we developed is able to change only the utility functions in the model. Even with this limitation, the method has to search a large solution space. Nevertheless, the results of experiments show that this simplest form of automatic revision managed to successfully adjust the experimental model to new data. Valid methods, able of performing more flexible revision types should be even more succesful, but at the cost of higher computational complexity.

4. REVISION METHOD

The approach we used to test the revision of decision models is described in section 4.1. In section 4.2, the proposed revision method is presented in detail.

4.1 Approach

We tested the idea of decision model revision in a specific and somewhat simplified setting. The models used were hierarchic qualitative decision models of DEX. We limited the type of revisions only to revisions of utility functions values. The hierarchy of the model and sets of attribute-values were therefore left unchanged. Limitations in revision types were necessary, since the amount of revision possibilities would be otherwise too large to manage even for small examples.

	price	*maintenance*	*ABS*	*size*	***car***
1	low	low	yes	big	very good
2	low	low	yes	middle	very good
3	low	low	yes	small	good
…					
53	high	high	no	middle	bad
54	high	high	no	small	bad

Table 1: Some examples of simulated data entries for the car selection model.

The new data consists of new alternatives (cars) described by all their basic attributes and the goal attribute (Table 1). Usually such data is collected from past decision problems. In our case, we obtained the data through simulation using a changed decision model. Some utility functions were altered in the original model, then all the possible combinations of basic attributes were evaluated with the altered model and each resulting goal attribute was added to its combination. This procedure provided simulated new data (Table 1) that reflected the changes in utility functions of the original model.

4.2 The Revision Method

The goal of our revision method is to update the given original model according to the new data that is not necessary coherent with the model. Original model and new data are inputs to the method. The output is a new model, which has the same structure as the original one, but its utility functions are adapted to the new data.

The revision procedure is iterative. For each entry in the new data set (alternative), we check whether its goal attribute (evaluation) matches the goal attribute given by original model for the same alternative. If the evaluation matches, we proceed to the next entry. If it does not, this means we have a clash, an incoherence between new data and the original model. When a clash is detected, we try to revise the utility functions of the original model to become consistent with the clashing data entry. The best single change (if it exists) of utility functions of one of the attributes is found and performed. The procedure tries to change the utility function of the goal attribute first, then the changes are attempted to the utility functions of the lower aggregate attributes in the hierarchy. A change is attempted only if it contributes to the model in the way that solves the clash, but for the change to actually happen, it has to comply with the rules of monotonicity. If the potential change causes the utility function to become non-monotone, it is rejected.

```
revise(origModel, newData):
  newModel = NONE
  origCA = getCA(origModel, newData)
  bestCA = origCA
  for d in newData:
    if clash(origModel, d):
      while changesPossible:
        change = find useful change
        if monotoneOK(change):
          CA = getCA(origModel, change, newData)
          if CA > bestCA:
            bestCA = CA
            bestChange = change
      if bestCA > origCA:
        newModel = makeChange(origModel, bestChange)
  return newModel
```

Figure 3: Pseudo-code of the proposed revision method.

Each individual change is evaluated with the classification accuracy (CA) of a changed model on the new data. If the CA is higher than CA of unchanged model on the new data and the CA of the model with the currently best change for solving the current clash, the change is recorded as currently best change and its CA becomes the new reference for other individual changes. When all the possible changes of the model to comply with the clashing data entry have been tested, the best change (the one with highest CA on new data) is perfomed. It is possible that such change does not exist or that there is no change that would solve the clash without producing more clashes. In that case the original model is left unchanged and the clash remains unsolved. The pseudo-code of the revision process is presented in Figure 3.

The presented revision method is simple, however there are many situations where a different solution could be made or at least debated about. The main drawback of the method is its dependance on the ordering of data entries in the new data set. When one clashing entry is satisfied with a change in the model, there is a chance that another clashing entry will not be satisfied because of the change that was performed when satisfying the first one. All possible changes could violate the monotonicity, because of the first change, for instance. This can be solved with a method that calls the revision with all the possible orderings of the clashing new data entries, but this process is computationaly demanding ($O(n!)$) and perhaps a heuristic or a greedy approach would be a better solution. Making changes that solve more clashes first is one of the possible heuristics.

Another possible improvement of the method would be the ability to make a series of changes simultaneously in one or many different utility functions. The computational complexity of such improvements would increase immensely, but so would the possible solution space.

5. EXPERIMENTS

5.1 Experimental Setting

The proposed revision method and a basic tool for the construction of DEX decision models were implemented in Python programming language. A simple decision model (Figure 1, Figure 2) was constructed as a representation of an original decision model made by the experts. Some changes were made to utility functions in this model (Figure 4) to represent the new model, the one that reflects the actual state of decision problem, but we are not aware of. In our case the changes were made so as to emphasize the importance of "costs" and inside "costs" to emphasize the importance of "price" (see underlined entries in Figure 4). The new model was used to generate the new data as described in section 4.1, 54 new data entries were obtained this way. Only the new data was available to the revision method.

price	*maintenance*	***costs***
low	low	low
low	middle	low
<u>low</u>	<u>high</u>	<u>low</u>
middle	low	low
middle	middle	middle
<u>middle</u>	<u>high</u>	<u>middle</u>
<u>high</u>	<u>low</u>	<u>high</u>
high	middle	high
high	high	high

ABS	*size*	***safety***
no	small	bad
no	middle	acceptable
no	big	good
<u>yes</u>	<u>small</u>	<u>acceptable</u>
yes	middle	good
yes	big	very good

costs	*safety*	***car***
low	very good	very good
low	**good**	**very good**
low	**acceptable**	**good**
<u>low</u>	<u>bad</u>	<u>acceptable</u>
middle	very good	good
middle	good	good
<u>middle</u>	<u>acceptable</u>	<u>middle</u>
middle	bad	bad
high	**very good**	**middle**
<u>high</u>	<u>good</u>	<u>acceptable</u>
high	acceptable	bad
high	bad	bad

Figure 4: Changed utility functions for our example case. The rows that were first changed are underlined. The rows that were additionally changed are printed in bold.

The performance of the revision method was measured as CA of the revised model on new data. In that way it can be seen how well the revised model reflects the new (or actual) state of decision problem. We made two types of experiments. In the first one we measured CA on the whole new data set, whereas in the second, the new data set was divided into a training and a test set. The training set was used by the revision method and the test set was used to test the revised model. The first type of measurement of CA shows if the method works, that is, if it succeeds to adjust the model to data. However, it is usually undesirable for the model to fit to training data too tightly, as the data can be noisy and can cover only a part of problem space. In our setting this problem is less severe, because the hierarchy and values sets are fixed and the possible changes to the model are limited.

Nevertheless, we made the second type of measurements too, using cross-validation, to get an impression of the usefulness of the revision method in real world problems.

5.2 Experiments

The first alteration of the original decision model is represented as underlined rows of utility functions in Figure 4. We measured CA of the revised model on new data (obtained from changed model) in three ways:

- all data entries were used for revision (training) and for testing
- 10 fold cross-validation
- 5 fold cross-validation

The cross-validation procedure is a classic evaluation procedure, extensively used in machine learning community (Mitchell 1997). We divide data into *n* folds and then use *n–1* folds for training (in our case for revision) and the remaining one for testing and obtaining measurements. The process of training and testing is repeated *n*-times, each time using different *n–1* folds for training and a different one for testing. At the end, the results of all the *n* tests are averaged. Usually 10 or 5 folds are used.

evaluation method	original model	revised model
all data entries	57.4 %	83.3 %
10 fold cross-validation	56.3 %	68.0 %
5 fold cross-validation	56.9 %	69.8 %

Table 2: Experimental evaluation of revised model in comparison to original model regarding CA on data from the changed model.

The results of the first experiment are shown in Table 2. The substantial improvement of CA on training data indicates that the proposed revision method succeeds in adapting the model to a situation described with new data. Results obtained with cross-validation indicate that improvement in real-world situation could be somewhat worse, but still considerable.

Another set of changes was added to the changed model, to test the method with another setting. Additional changes are presented as bold printed rows in Figure 4. As the model was additionally changed in the direction we mentioned in section 5.1, even better results were expected. Indeed, further improvement can be seen in Table 3.

evaluation method	original model	revised model
all data entries	31.5 %	90.7 %
10 fold cross-validation	30.7 %	74.3 %
5 fold cross-validation	31.5 %	70.0 %

Table 3: Experimental evaluation of revised model in comparison to original model regarding CA on data from the additionally changed model.

Results on both artificial data sets indicate that the method is valid and that it could be useful in real-world problems as well. An interesting phenomenon is the difference of results between 10 and 5 fold cross-validation (CV). In the experiment with the first data set, the result of 5 fold CV is slightly better, but in experiment with the second data set, the 5 fold CV result is worse than the one of 10 fold CV. This could be explained with the scope of changes in the original model and the portion of learning data entries. In the first experiment, there were only few changes made to the original model and the majority of data set entries were still in accordance with the original model (the CA of original model is higher than 0.5). With a larger learning data set (10 fold CV), more changes were made to the original model during revision, but the small test data set did not reflect those changes substantially. The 5 fold CV had a better CA, because the learning data set was smaller (fewer changes were made during revision) and the test data set was larger, hence reflected the direction of changes better. The same can be observed in the second experiment, however in this case the situation is turned around. The majority of data entries reflect the changes in a way that the original model does not (the CA of original

model is lower than 0.5), hence the evaluation method that provides more learning data entries (10 fold CV), yields a better evaluation result than the evaluation method with less data entries.

To test a hypothesis about the dependence among the scope of changes (the amount of clashing data entries), proportion of learning data entries in evaluation and the result of evaluation, we conducted another experiment. We evaluated the revision method on both data sets with 3 fold CV. The result of this evaluation is shown in Table 4. For easier comparison, the previous two CV evaluations are shown in Table 4, too.

Evaluation method	Changed model data		Additionally changed model data	
	Original model	Revised model	Original model	Revised model
10 fold cross-validation	56.3 %	**68.0 %**	30.7 %	**74.3 %**
5 fold cross-validation	56.9 %	**69.8 %**	31.5 %	**70.0 %**
3 fold cross-validation	57.4 %	**74.1 %**	31.5 %	**63.0 %**

Table 4: Results of revision method evaluation from three types of cross-validation.

The results of 3 fold CV support our conclusions about differences among types of CV used. This suggests that when the new data are not very inconsistent with the original model, the scope of changes in revision should be low and when the new data are very inconsistent with the original model, the scope of changes in revision should be higher.

6. CONCLUSIONS

The revision of decision support models is a common task that has not been properly automated by decision support tools yet. We provided some insight into the general problem of decision model revision with the focus on revision of hierarchical qualitative decision models. A possible taxonomy of automatic revision methods with some guidelines was provided along with motivation for their development.

A simple revision method that alters only utility functions of qualitative hierarchical multi-attribute models was proposed and thoroughly described in the paper. The proposed method was implemented and experimentally evaluated using a simulated data set. In all experiments, the classification accuracy of the revised model measured on the new data set has improved and the changes of the model correctly reflected the simulated changes in the decision environment. The results prove that the method is valid and indicate that it could be succesfully used in practice.

For further work, we plan to evaluate the proposed revision method in a real-world setting. Some improvements are planned with the use of heuristic and greedy approaches. It would be interesting to experiment also with automation of other revision types, to explore their feasibility and usefulness. We hope our work and experimental results will serve as motivation for further efforts in development of automatic revision methods, to make the use of decision models more simple, more useful and more frequent.

REFERENCES

Bohanec, M., & Rajkovič, V. (1990) DEX : An Expert System Shell for Decision Support. *Sistemica*, 1 (1), pp.145-157.

Bohanec, M., & Rajkovič, V. (1999) Multi-Attribute Decision Modeling: Industrial Applications of DEX. *Informatica*, 23 (4), pp.487-491.

Bohanec, M., Rajkovič, V. & Cestnik, B. (2003) Five Decision Support Applications. IN: Mladenić, D., Lavrač, N., Bohanec, M. & Moyle, S. ed. *Data Mining and decision support : integration and collaboration, (The Kluwer international series in engineering and computer science, SECS 745)*. Boston; Dordrecht; London, Kluwer Academic Publishers. pp.177-189.

Bureau of Resource Sciences (1997) A radioactive waste repository for Australia: Site selection study - Phase 3: Regional assessment: A public discussion paper. Bureau of Resource Sciences, Canberra.

Clemen, R. T. (1996) Making Hard Decisions: an Introduction to Decision Analysis, Wadsworth Publishing Company.

Greenberg, M., Burger, J., Powers, C., Leschine, T., Lowrie, K., Friedlander, B., Faustman, E., Griffith, W., & Kosson, D. (2002) Choosing remediation and waste management options at hazardous and radioactive waste sites. *Remediation*, 13 (1), Wiley InterScience. pp.39-58.

Han, J., & Kamber, M. (2001) Data mining: concepts and techniques, Morgan Kaufman Publishers.

Keeney, R. L., Raiffa, H., & Meyer, R. (1993) Decisions with Multiple Objectives: Preferences and Value Trade-offs, Cambridge Univ Press.

Michalowski, W., Rubin, S., Slowinski, R., & Wilk, S. (2003) Mobile clinical support system for pediatric emergencies. *Decision Support Systems*, 36 (2), Elsevier Science. pp.161-176.

Mitchell, T.M. (1997) Machine Learning, New York: McGraw-Hill.

Saaty, T. L. (1980) The Analytic Hierarchy Process, McGraw-Hill.

Triantaphyllou, E. (2000) Multi-Criteria Decision Making Methods: A Comparative Study. Kluwer Academic Publishers.

Turban, E., & Aronson, J. E. (2001) Decision Support Systems and Intelligent Systems, Prentice Hall.

Zupan, B., Bohanec, M., Demšar, J., & Bratko, I. (1999) Learning by discovering concept hierarchies. *Artificial Intelligence*, vol. 109, pp.211-242.

Zupan, B., Porenta, A., Vidmar, G., Aoki, N., Bratko, I., Beck, JR. (2001) Decisions at Hand: A Decision Support System on Handhelds, Medinfo 2001, London, UK.

COPYRIGHT