Smart-Swaps – A decision support system for multicriteria decision analysis with the even swaps method

Jyri Mustajoki* and Raimo P. Hämäläinen

Helsinki University of Technology

Systems Analysis Laboratory

P.O. Box 1100, FIN-02015 HUT, Finland

E-mails: jyri.mustajoki@hut.fi, raimo@hut.fi

*Corresponding author: Tel. +358-9-451 3065, fax +358-9-451 3096

Abstract: This paper introduces a new web-based decision support tool called Smart-Swaps to support multicriteria decision analysis. The decision maker's preferences are elicited with the even swaps method, which is an elimination process based on value trade-offs. The software provides a platform for carrying out the elimination process and implements a preference programming model to give suggestions to the decision maker on how to proceed with the process. Such decision support can provide substantial help to the decision maker, especially when the number of alternatives and attributes is large.

Keywords: multicriteria decision analysis, multicriteria decision support systems, even swaps, preference programming, trade-offs

1. Introduction

Multicriteria decision analysis (MCDA) is a structured approach to analyze problems with several criteria and alternatives. It helps the decision maker (DM) to make consistent decisions by taking all the important objective and subjective factors into account. In their book *Smart Choices*, Hammond, Keeney and Raiffa [11] suggest following the PrOACT (an acronym for **P**roblem, **O**bjectives, **A**lternatives, **C**onsequences and **T**rade-offs) working phases to consistently carry out the MCDA process. They also present the *even swaps* method for preference elicitation (see also Ref. [10]). It is an elimination process based on value trade-offs (see e.g. Refs. [18, 20]) which are called even swaps. In an even swap, the DM changes the consequence of an alternative on one attribute, and compensates this change with a preferentially equal change in the consequence of another attribute. This creates a new virtual alternative with revised consequences. This is as preferred as the initial one, and thus it can be used as a surrogate, even though it is not a real alternative.

The aim of the even swaps process is to carry out even swaps that either make attributes irrelevant or alternatives dominated. An attribute is irrelevant if all the alternatives have equal consequences on this attribute. Alternative x dominates alternative y if x is better than or equal to y on every attribute and better at least on one attribute. Irrelevant attributes and dominated alternatives can both be eliminated, and the process continues until only the most preferred alternative remains. The concept of practical dominance is also introduced [10]. Alternative x practically dominates alternative y if y is slightly better than x on only one or few attributes but x clearly outranks y on several other attributes. Thus, y can be eliminated in order to reduce the problem in obvious cases without a need to carry out unnecessary even swap tasks.

Even swaps is a conceptually simple process intended for the general audience. The DM does not need to have a mathematical or decision analytical background to use the method. For example, the DM does not have to explicitly define the preferences over the attributes in general or to make any assumptions about the form of a value function [4]. On the other hand, in the even swaps process, the

interpretation of the results is not as transparent as in traditional multiattribute value tree (MAVT) approaches. For a comparison between even swaps and MAVT, see Ref. [2].

In this paper, we introduce a new web-based Smart-Swaps software [9] to support the MCDA process with the even swaps method. We focus on discussing the new opportunities that computer support can provide. Especially from the viewpoint of carrying out the even swaps elimination process, such software can be of substantial help. For example, procedural support of Smart-Swaps for dynamically managing the consequences table can be very useful, as the table continuously changes during the process. The software also implements a new method of Mustajoki and Hämäläinen [25] to help the DM to identify practical dominances and to find suitable candidates for the next even swaps. This method is based on the preference programming [1, 31, 32], which is an MAVT approach that allows modeling of incomplete information in the DM's preferences with intervals. Smart-Swaps is available for on-line academic use at <http://www.smart-swaps.hut.fi> and it is also a new tool in the family of MCDA software available on the Decisionarium site for decision support (<http://www.decisionarium.hut.fi>) [7, 8].

In spite of its simplicity, it seems that the even swaps method has rarely been applied in practice. To our knowledge, the only reported applications in literature are the ones in strategy selection in a rural enterprise [16], in environmental planning [6] and in health care [22]. In addition, the use of the method has been demonstrated in military options analysis [13] and in risk valuation [35]. The lack of real applications can be partly due to the fact that, so far, there has not been any software to support the approach. We hope that the introduction of the Smart-Swaps software described here will make the approach more popular in real life, too.

This paper is organized as follows. Section 2 gives a brief introduction to multicriteria tools. Section 3 discusses the opportunities of computer support in the different ProACT working phases and describes the Smart-Swaps software in terms of how these opportunities are adapted in the software. An example is given in Section 4 and Section 5 discusses the use of the software in practice. Section 6 concludes the paper.

2. Multicriteria decision support systems

Multicriteria decision support systems (MCDSS) or tools are computer-based interactive software designed to support the modeling of multicriteria problems with MCDA methods. The aim is to provide help for structuring the problem, eliciting preferences and analyzing the results, so that the DM can focus on the core of the problem while the technical issues are taken care of by the computer.

Recent development in computer technology has provided new opportunities to enhance the use of MCDA [3, 21, 28, 33]. For example, with today's multimedia facilities, MCDA methods can be visualized to help preference elicitation and the analysis of the results. Increased computational capabilities have also made it possible to create interactive software with new computationally demanding methods, such as linear programming. In recent years, the proliferation of the World Wide Web has enabled MCDSSs that are offered and maintained in one location and are still easily available for remote use.

There are various MCDSSs available (see e.g. Ref. [14]). For recent reviews of these see, for example, Refs. [5, 23, 34]. The software range from customized tools for some specific application (see e.g. Refs. [15, 27]) to general purpose systems providing a choice between many different MCDA methods as well as group decision support facilities (see e.g. Ref. [24]). Some of the software allow the use of trade-offs as one of the weighting methods but, to our knowledge, Smart-Swaps is the first software that implements the even swaps method.

3. The Smart-Swaps decision support tool

Table 1 shows an overview of the support provided by Smart-Swaps in the different PrOACT working phases. To support the stepwise nature of the PrOACT process, the user interface for the process management is implemented with tab panels that follow the phases of the process (Figure 1). The first three phases (*Problem*, *Objectives* and *Alternatives*) are on the same panel, as active interaction between these is often needed but the *Consequences* and *Trade-offs* phases are on their own panels. The aim is to give the DM a clear indication of the course of the process but, at a same

time, allow him/her to easily return to the earlier phases of the process, if necessary. The example presented in the figures is the Alan Miller's office selection problem adapted from Refs. [10, 11], and it is described in more detail in Section 4.

3.1. Problem, objectives and alternatives

The first phase of the process is to set a problem framework in which the decision will be made. Consideration of the problem from different perspectives is of essential importance to get a view of what the problem really is about. The next phases are the identification of objectives and alternatives. Objectives define what the DM wants to achieve, and alternatives describe the possible actions that can be taken. There are different procedures for identifying these. For a related discussion, see Refs. [11, 17].

For each of these phases, the Smart-Swaps software provides a section in which the DM can write down the problem description and list and order the objectives or alternatives (Figure 1). However, the greatest challenge in these phases is to help the DM to take all the different viewpoints into account and still construct a practically useful and concise model. For example, providing information on how to avoid possible biases in the modeling is one way of guiding the DM. In Smart-Swaps, the methodological guidance is implemented with help facilities providing detailed information for all the phases of the process. For details, see Section 3.4.

One should note that the order of these first three phases is not fixed and the most creative ideas are usually found by considering the phases interchangeably [11]. For example, a thorough analysis of the alternatives can bring in new views that put the problem into a completely new perspective. In such cases, the DM should go back and reframe the problem according to this new framework.

3.2. Initialization of the consequences table

In the *Consequences* phase, the DM creates a consequences table in which the performance of each alternative is measured with respect to each attribute (Figure 2). Sometimes the attributes for

measuring the achievements of the objectives are natural ones (e.g. minimizing the costs) but in some cases they have to be constructed [19].

Scales for the attributes can either be discrete or continuous, and Smart-Swaps supports both of these. For continuous scales, the DM can use any decimal number to represent the performance levels of the attributes associated with each alternative. For discrete scales, the DM can use either one of the predefined scales (e.g. "Very Good – Good – Intermediate – Bad – Very Bad"), or create a scale of his/her own. This opportunity for customizing the scale allows the DM to construct scales indicating the preference order for any set of consequences. The DM can, for example, create a preference order for a set of numerical consequences, in which the DM's preference of the consequences increases non-monotonically.

In practice, the DM directly enters the consequences into the consequences table. One should, however, note that the attributes in the table represent the objectives defined during the *Objectives* phase. Thus, the DM should make sure that, for each objective, there is a corresponding measurable attribute.

3.3. Support for the even swaps process

The final PrOACT working phase is to elicit the preferences of the DM. Both in the Smart Choices book and in the Smart-Swaps software, the even swaps method is used for this task but in principle any other MCDA method could be used as well. In Smart-Swaps, the interface for supporting the even swaps elimination process includes a panel showing the current consequences table and buttons for the possible actions (Figure 3). There is also an information area in the upper right corner of the panel showing any additional information that may be useful to the DM during the process (Figure 4).

As a result of the process, the DM ends up with the most preferred alternative. However, with respect to transparency and justifiability of the result, it is important to understand how this conclusion has been reached. Smart-Swaps documents the whole process by keeping log of the actions made by the DM during the process. This history of log saves all the information about the

swaps made by the DM and about the eliminated attributes and alternatives, as well as the state of the consequences table after each swap (see Figure 5).

The DM can undo and redo the actions that have been made. The DM can, for example, return to the beginning of the process to check whether the new virtual alternative with the revised attribute values indeed is of equal value to the initial alternative. The DM can also backtrack to some intermediate point of the process and restart from this point by following another sequence of swaps to see whether he/she ends up with the same final solution. This makes it possible to carry out sensitivity analyses in terms of studying whether different sequences of swaps produce different final results.

3.3.1. Making an even swap

To make an even swap, the DM first selects the three cells in the consequences table between which he/she wants to carry out an even swap. These include (i) a cell in which the consequence change is made, (ii) a reference cell indicating the value against which this consequence is traded, and (iii) a cell in which the change is compensated (striped cells in Figure 4). Once these have been selected, the software informs the DM what can be achieved with this swap, that is, which alternatives can possibly become dominated and which attributes irrelevant. For example, Figure 4 informs that both *Baranov* and *Lombard* might become dominated by making a swap in which a change in *Office Size* of *Montana* from 950 to 700 is compensated in *Monthly Cost*. This information can be very useful, as otherwise it might be difficult to see which alternatives may become dominated. By changing the selected cells, the DM can easily study the implications of different possible swaps before confirming the final swap to be made.

The actual even swap is defined in a separate dialog window. The DM is asked to define which consequence change on attribute j of alternative $\mathbf{x}$ would compensate the given consequence change on attribute i (Figure 6). The phrasing in the dialog window helps the DM understand whether he/she should decrease or increase the current consequence. However, if the DM accidentally makes a swap into the wrong direction, the software informs him/her about this and requests to redefine the swap.

3.3.2. Identification of irrelevant attributes and dominated alternatives

After each swap, Smart-Swaps automatically identifies irrelevant attributes and dominated alternatives. Especially in detecting the dominance relations, the DM can benefit from this support, as this task requires comparing all the possible pairs of alternatives. According to the basic idea of the method, irrelevant attributes and dominated alternatives should be eliminated. However, the software asks the DM to confirm these eliminations, as this helps the DM to fully understand the process and the reasons for elimination. The software also provides an option to retain the irrelevant attributes in the consequences table so that they are marked as eliminated. This may help to keep the big picture in mind and contribute to the overall understanding of the process.

3.3.3. Identification of practically dominated alternatives

Practically dominated alternatives are typically even harder to identify than dominated ones. Smart-Swaps uses preference programming to identify candidates for these [25]. The idea of this approach is to use an additive MAVT model in parallel with the even swaps process to model the preferences of the DM. At the start of the process, the weight ratios between the attributes are unknown but it is reasonable to assume some general bounds for these. Preference programming is applied to set these bounds. The pairwise dominance concept (see e.g. Refs. [12, 31, 36]) is then applied to identify practically dominated alternatives, as any alternative that is dominated in a pairwise sense in this model can be considered as a candidate for being practically dominated.

In this model, the overall value of an alternative described by the consequence vector $\mathbf{x} = (x_1, \dots, x_n)$ is obtained with an additive value function

$$v(\mathbf{x}) = \sum_{i=1}^n w_i v_i(x_i) \quad (1)$$

where n is the number of attributes, x_i is the consequence of this alternative with respect to attribute i , $v_i(x_i)$ is its score on $[0, 1]$ scale, and w_i is the weight of attribute i representing the relative importance of this attribute. The weights are normalized to sum up to 1.

The bounds for the attributes' weights are defined by setting a general upper bound $r \geq 1$ for the weight ratios, which leads to a set of constraints

$$\frac{w_i}{w_j} \leq r, \quad \forall i, j=1, \dots, n, i \neq j. \quad (2)$$

These constrain the feasible region of the weights S . Similar bounds are also set for the attribute scores. As a result, the overall values of the alternatives will be ranges of possible values rather than pointwise values. During the even swaps process, the new preference information obtained from the given swaps is applied to derive new tighter constraints in the model, and consequently to make the model more precise. For details, see Ref. [25].

Preference relations between the alternatives are studied with dominance concepts. Alternative $\mathbf{x}$ dominates alternative $\mathbf{y}$ in a pairwise sense if the overall value of $\mathbf{x}$ is at least as high as the overall value of $\mathbf{y}$ with every feasible combination of the weights and the ratings, that is, if

$$\min_{\mathbf{w} \in S} \sum_{i=1}^n w_i [\underline{v}_i(x_i) - \bar{v}_i(y_i)] \geq 0, \quad (3)$$

where S is the feasible region of the weights, $\underline{v}_i(x_i)$ and $\bar{v}_i(y_i)$ are the lower and upper bounds for $v_i(x_i)$ and $v_i(y_i)$, respectively, and the inequality is strict at least for some $\mathbf{w} = (w_1, \dots, w_n) \in S$. In other words, there are no such feasible weights and attribute scores with which alternative $\mathbf{y}$ would have a higher overall value than alternative $\mathbf{x}$. Thus, alternative $\mathbf{y}$ can be considered as a candidate for being practically dominated, as the bounds for the feasible weights and scores are initially set so that they include all the values that are acceptable in practice. For example, the default value of r in (2) is 5, but the bounds can also be adjusted by the DM. The tighter the bounds are, the more candidates for practical dominance the software will provide. One should, however, note that if the bounds are too tight, some of the suggestions may not be correct.

In practice, the candidates for practically dominated alternatives are marked with corresponding labels (see e.g. *Parkway* in Figure 3). Clicking of the labels opens a comparison dialog in which the candidate for a practically dominated alternative is presented side by side with an alternative

dominating it (Figure 7). The reasoning for the practical dominance is explained, and based on this information the DM is asked to confirm whether the alternative should be eliminated.

After each swap, the software checks whether the new weight ratio constraint derived from this swap is conflicting with any other constraints in the model. If this is the case, the software informs the DM that the given statement might be inconsistent with some earlier one and suggests checking the consistency of the given swaps by backtracking the process. If the DM does not find any inconsistencies between the given swaps, the inconsistency in the model is likely to originate from excessively strong assumptions of the model. For example, the value function may not truly be an additive one. Then, the software cannot naturally use the conflicting constraints in the model but only the initial constraints are used. However, as the assumptions of the model were just found to be invalid, any subsequent suggestions can not be assumed to be very accurate either.

3.3.4. Suggestions for the next swap

In a typical case, there are numerous possibilities for the next swap, and it can be very difficult to decide which one of these should be carried out next. Hammond, Keeney and Raiffa [11] provide practical advice for selecting the next swap, for example, they suggest making the easiest swaps first (e.g. on attribute money). On the other hand, the idea of the even swaps process is to carry out such swaps that make attributes irrelevant or alternatives dominated. However, in practice the identification of such swaps can be excessively difficult, especially in problems with several attributes and alternatives.

In Smart-Swaps, the DM has an option to let the software suggest suitable candidates for the following swap. The software scans through the problem and calculates the efficiency of each possible swap by calculating the minimum number of swaps needed after this swap to make any attribute irrelevant or any alternative dominated. Based on this number, the software creates and continuously updates lists of suitable candidate swaps both for making attribute(s) irrelevant and alternative(s) dominated. The DM can ask the software to suggest the next swap from either of these lists by clicking the corresponding button (Figure 4).

By default, the software creates the list of candidate swaps for making attribute(s) irrelevant in the following way.

- Step 1. The software identifies all the swaps with which any attribute would become irrelevant. These form the initial list of suggestions. If there are no such swaps, the software lists those swaps that could make some attribute to be one swap or two swaps etc., apart from being irrelevant.
- Step 2. The obtained list is ordered according to the applicability index of each swap to reach dominance [25]. The ordering is needed as there are typically many sequences of swaps that lead to dominance or irrelevance with the same number of swaps. The applicability index is described in detail below.
- Step 3. Of the swaps involving the same two attributes and two alternatives, all but the first one are excluded from the list. The reason is that once one of these has been identified, the others can usually be easily identified, too.
- Step 4. All but the first six swaps are excluded from the list, as giving too many options may complicate the DM's task rather than facilitate it.

The list of candidates for making alternative(s) dominated is created analogously.

The applicability index indicates how large a consequence change the DM is allowed to make in the compensating attribute in order to reach dominance, in relation to the estimated consequence change in this attribute. Mathematically, the index is calculated from the preference programming model as follows. Assume that the DM's preferences are modeled with preference programming and alternative $\mathbf{y}$ outranks alternative $\mathbf{x}$ only on attribute i . Then, the DM could try to make $\mathbf{y}$ dominated by $\mathbf{x}$ by carrying out an even swap in which a change in attribute i of alternative $\mathbf{x}$ from x_i to y_i is compensated with a change in attribute j from x_j to x'_j . The applicability index for this swap to make alternative $\mathbf{x}$ dominate alternative $\mathbf{y}$ is

$$d(\mathbf{x} \rightarrow \mathbf{y}, i, j) = \min\left(\frac{v_j(x_j) - v_j(y_j)}{(w_i / w_j)[v_i(y_i) - v_i(x_i)]}\right), \quad (4)$$

where the extremes for the weight ratio w_i / w_j and for the rating differences are obtained from the constraints on the preference programming model. The higher applicability index value a swap has, the more likely it is that the consequence x'_j would be above y_j after this swap. Consequently, the more likely a dominance relation would be reached with this swap. Similar indices can be calculated for cases where alternative $\mathbf{y}$ outranks alternative $\mathbf{x}$ in two or more attributes. For details of calculating the applicability index, see Ref. [25].

When the DM asks for a suggestion, the software gives him/her the next suggestion on the corresponding list by highlighting those cells of the consequences table that are involved in the swap. The software informs the DM what can be achieved with this swap to help the DM see the logic behind the suggestion. Now, the DM needs to choose whether to make this swap or ask the software to give the next suggestion on either of the suggestion lists. Naturally, the DM can also select the swap by him/herself, if he/she prefers to make a swap, for example, on some easily measurable attribute. Once all the suggestions on the list are given to the DM, the software returns to the first one.

There are also several options for suggesting the swaps. The DM can, for example, choose whether or not to show the applicability index value for each swap. The DM can also choose that in Step 2 the list of suggestions is first ordered according to the number of different alternatives that can be made dominated by each swap and, if there are ties on this number, then according to the applicability index.

3.3.5. *Rankings table*

A rankings table shows the attribute-wise rankings of the alternatives and it can be applied to get an overview of the overall performances of the alternatives [11]. In Smart-Swaps, the DM can switch between the two views showing the consequences table and the rankings table. In addition, the software provides an option to visually indicate the rankings of the alternatives by a color coding on the cells of the consequences table. The best alternative with respect to each attribute is shown in white and the worst alternative in yellow. The colors of the other alternatives follow a linear white–yellow color scale so that the lower ranking the alternative has, the darker shade of yellow there is. In

this way, the DM can easily see the attribute-wise rankings of the alternatives from the consequences table.

3.3.6. Computational issues

The preference programming model requires using linear programming to solve the related optimization problems. Since there are constraints only on the pairwise ratios of the attribute weights, the graph based algorithm of Salo [30] can be used to quickly solve these problems. This algorithm is also implemented in Smart-Swaps. For example, in a case of eight attributes and twelve alternatives [29], the identification of dominances with Smart-Swaps takes less than two seconds with a 2.4 GHz computer. Yet, in larger problems the graph algorithm is not very efficient. Thus, the software uses a basic simplex algorithm in problems with ten or more attributes.

3.4. The help facilities

The PrOACT process is intended to be available for non-experts too which means that good help facilities are needed. In Smart-Swaps, the help pages can be configured to follow the DM during the process so that the help screen is automatically updated to show the relevant information. This feature can be very valuable for an inexperienced DM carrying out the process for the first time. The first three phases are on the same panel window, and the help system automatically refers to that section of the panel which is under consideration.

There is help separately for theory and for practice (Figure 8). The theory section explains the theoretical background of each task and gives guidance on what should be taken into account in the task. The practice section provides detailed information on how to carry out the current task.

3.5. Model management

The software allows saving the models on the server computer. Thus, one can open models from different Internet locations without the need to transfer the models to remote computers. For each model, the process history will also be saved. This allows the DM to later analyze the process and

study path dependence by comparing the results of the processes with some other sequences of even swaps.

The server provides a public folder or the DM can create a private password-protected working folder for him/her. For beginners, there are also sample models available including those described in the Smart Choices book.

4. Example

We illustrate the opportunities to support the even swaps process with the Alan Miller's office selection problem considered in Refs. [10, 11] (Figure 3). The first phases of the PrOACT process are quite straightforward and, thus, we only demonstrate the support provided for the even swaps elimination process. In the example, the DM wants to minimize the consequence levels of *Commute time* and *Monthly cost* and maximize the level of the other attributes.

At the start of the even swaps process, the software scans through the consequences table and suggests eliminating *Pierpoint*, as it is dominated by *Lombard* (Figure 3). Thus, the DM eliminates it. The software also identifies *Parkway* as practically dominated by *Montana*. The DM carefully compares these alternatives side by side (Figure 7) and notices that *Montana* outranks *Parkway* on each attribute except in *Monthly cost*, in which it is only 50 dollars weaker. Based on this comparison, the DM decides to eliminate *Parkway*.

Since there are no more dominated alternatives, the DM continues the process by making an even swap. However, he first asks the software to give suggestions for a swap that would make attributes irrelevant. Smart-Swaps provides three suggestions to make *Commute time* irrelevant by changing *Commute time* of *Baranov* from 20 to 25 and by compensating this change in *Office size*, *Office services* or *Client access*. Each of these swaps seems to be quite reasonable and the DM makes a swap in which the change in the *Commute time* of *Baranov* from 20 to 25 is compensated with a change in *Client access* from 70 to 78. Consequently, *Commute time* becomes irrelevant, and is eliminated (Figure 4).

The consequences table changes with this swap and the DM can ask for new suggestions for the next swap. Now, no attribute can be made irrelevant with a single swap but any one of the attributes can be made irrelevant with two swaps. Thus, no suggestions for making an attribute irrelevant are given. However, when asking for a suggestion to make an alternative dominated, the software suggests six possible swaps. One should note that there are in fact 36 such swaps but only the ones with the highest applicability index values are given. The first suggestion is a swap in which *Monthly cost* is used to compensate a change in *Office size* of *Montana* from 950 to 700 (Figure 4). With this swap, both *Lombard* and *Baranov* can become dominated. However, by evaluating this swap the DM notices that it is not easy to make both alternatives dominated, as this would require that the consequence of *Montana* in *Monthly cost* was chosen to be below 1500. Nevertheless, the DM makes this swap so that the suggested change in *Office size* of *Montana* is compensated with a change in *Monthly cost* from 1900 to 1650. This results in the elimination of *Lombard* as a dominated alternative but *Baranov* still remains non-dominated.

When asking for a new even swap suggestion for making an alternative dominated, the software suggests a change in *Office services* of *Montana* from A to C and compensation in *Monthly cost*. The DM makes this swap with compensation in the *Monthly cost* of *Montana* from 1650 to 1350. As a result, *Baranov* is eliminated, and *Montana* is the most preferred alternative.

To see whether the result is sensitive to changes in the sequence of the swaps, the DM restarts from the situation in which *Pierpoint* and *Parkway* were eliminated (Figure 3 without *Pierpoint* and *Parkway*). Initially, the DM carried out a swap that made *Commute time* irrelevant but now he asks the software to suggest a swap that could make some alternatives dominated. There are three suggestions with which *Lombard* could become dominated by *Montana* by making an even swap between *Monthly cost* and either *Office size*, *Office services* or *Client access*. Of these, the DM selects a swap in which he changes *Monthly cost* of *Lombard* from 1700 to 1900 and compensates this with a change in *Office size* from 700 to 900, when *Lombard* becomes dominated by *Montana*. After this he makes two more swaps suggested by the software. These are the change in *Commute time* of *Baranov* from 20 to 25 and the compensation in *Office size* from 500 to 550, and the change in *Monthly cost* of

Montana from 1900 to 1500 and the compensation in *Office size* from 550 to 950. As a result, *Baranov* is eliminated and *Montana* is found to be the most preferred alternative also through this other sequence of swaps. This convinces the DM of the result.

5. Discussion

The example shows how the Smart-Swaps software can provide convenient help for the even swaps process. Besides procedural support in making even swaps, the software helps the DM by providing information about the efficiency of the available swaps and about the attributes and alternatives that can be easily eliminated. Although the even swaps method was originally designed to be an easy-to-use pen-and-paper tool, we believe that the introduction of this new software broadens the application area of the method to more complex problems. The larger the problem, the more useful the software is in the screening of all the different possibilities.

The software is designed to suit both individual DMs and decision analysts. The default settings for all the options are such that an inexperienced DM should be able to proceed with the process without resetting them. However, an advanced user or a decision analyst can change these options to obtain insights of the suggestions provided by the preference programming model. For example, by default the value of the applicability index is not shown, as it may confuse an inexperienced DM rather than help him/her, but an advanced DM or a decision analyst can choose to show it to get more detailed information about the estimated efficiencies of the suggested swaps.

We think that the main advantage of the software is that it takes care of the technical tasks of the process so that the DM can focus on the problem itself and on the thinking process related to his/her preference elicitation. The aim of the software is not to automate the process but to give helpful suggestions. The DM should use common sense because, for example, the swaps suggested by the software can sometimes be more difficult to carry out than swaps on easily measurable attributes. Nevertheless, the software never suggests what consequences the DM should enter in the swaps but the DM has to consider these by him/herself.

Examination of the impacts of software support on the efficiency of the process requires experiments with real users. Our preliminary tests with university students show that in large problems the DM can indeed considerably benefit from using the Smart-Swaps software [26]. However, a thorough evaluation based on real users remains a topic of future research.

6. Conclusions

Despite the superficial simplicity of the even swaps method, we believe that the procedure can benefit from computer support both in the PrOACT process and in making trade-offs and managing the consequences table during the even swaps process. We demonstrate how the Smart-Swaps software provides this support and show the effectiveness of computer support with the new preference programming approach.

Even swaps is a relatively new method and its applicability in practice remains to be demonstrated. Smart-Swaps is designed so that one should be able to use it without expert help. This should increase the number of real life applications. The software also introduces new attractive features into the method itself by allowing rapid backtracking as well as documentation of the steps in the process. These can be important features which are likely to increase the transparency of the process. These features also make interesting future research possible, for example, on the behavioral aspects including the role of different starting strategies and the possible path dependence of the results.

Acknowledgements

The authors acknowledge Pauli Alanaatu, Ville Karttunen, Arttu Arstila and Juuso Nissinen for their contributions in the programming of the Smart-Swaps software. Jyri Mustajoki acknowledges the financial support of the Academy of Finland (project 32641), and the Jenny and Antti Wihuri Foundation.

References

- [1] A. Arbel, Approximate Articulation of Preference and Priority Derivation, *European Journal of Operational Research* 43 (1989) 317-326.
- [2] V. Belton, G. Wright and G. Montibeller, When Is Swapping Better than Weighting? An Evaluation of the Even Swaps Method in Comparison with Multi Attribute Value Analysis, Department of Management Science, University of Strathclyde, Research Report 19/2005.
- [3] H.K. Bhargava, D.J. Power and D. Sun, Progress in Web-based Decision Support Technologies, *Decision Support Systems* (to appear).
- [4] J. Butler, D.J. Morrice and P.W. Mullarkey, A Multiple Attribute Utility Theory Approach to Ranking and Selection, *Management Science* 47(6) (2001) 800-816.
- [5] S. French and D.-L. Xu, Comparison Study of Multi-Attribute Decision Analytic Software, MCDM 2004 Conference, Whistler, B.C., Canada, August 6-11, 2004.
- [6] R. Gregory and K. Wellman, Bringing Stakeholder Values into Environmental Policy Choices: A Community-based Estuary Case Study, *Ecological Economics* 39 (2001) 37-52.
- [7] R.P. Hämmäläinen, Decisionarium – Global Space for Decision Support, Systems Analysis Laboratory, Helsinki University of Technology, 2000. <http://www.decisionarium.hut.fi>
- [8] R.P. Hämmäläinen, Decisionarium – Aiding Decisions, Negotiating and Collecting Opinions on the Web, *Journal of Multi-Criteria Decision Making* 12(2-3) (2003) 101-110.
- [9] R.P. Hämmäläinen, J. Mustajoki, P. Alanaatu, V. Karttunen, A. Arstila and J. Nissinen, Smart-Swaps – Smart Choices with Even Swaps, Computer Software, Systems Analysis Laboratory. Helsinki University of Technology, 2003. <http://www.smart-swaps.hut.fi>
- [10] J.S. Hammond, R.L. Keeney and H. Raiffa, Even Swaps: A Rational Method for Making Trade-offs, *Harvard Business Review* 76(2) (1998) 137-149.
- [11] J.S. Hammond, R.L. Keeney and H. Raiffa, Smart Choices, A Practical Guide to Making Better Decisions, Harvard Business School Press, Boston, MA, 1999.

- [12] G.B. Hazen, Partial Information, Dominance, and Potential Optimality in Multiattribute Utility Theory, *Operations Research* 34(2) (1986) 296-310.
- [13] W.J. Hurley and W.S. Andrews, Option Analysis: Using the Method of Even Swaps, *Canadian Military Journal* 4(3) (2003) 43-46.
- [14] INFORMS OR/MS Resource Collection: Computer Programs, Institute for Operations Research and the Management Sciences, 2000.
http://www.informs.org/Resources/Computer_Programs/
- [15] M.P. Johnson, Spatial Decision Support for Assisted Housing Mobility Counseling, *Decision Support Systems* 41(1) (2005) 296-312.
- [16] M. Kajanus, J. Ahola, M. Kurttila and M. Pesonen, Application of Even Swaps for Strategy Selection in a Rural Enterprise, *Management Decision* 39(5) (2001) 394-402.
- [17] R.L. Keeney, *Value-focused Thinking. A Path to Creative Decisionmaking*, Harvard University Press, Cambridge, MA, 1992.
- [18] R.L. Keeney, Common Mistakes in Making Value Trade-offs, *Operations Research* 50(6) (2002) 935-945.
- [19] R.L. Keeney and R.S. Gregory, Selecting Attributes to Measure the Achievement of Objectives, *Operations Research* 53(1) (2005) 1-11.
- [20] R.L. Keeney and H. Raiffa, *Decisions with Multiple Objectives. Preferences and Value Tradeoffs*, John Wiley & Sons, Inc., New York, 1976.
- [21] D. Liu and T.J. Stewart, Integrated object-oriented framework for MCDM and DSS modeling, *Decision Support Systems*, 38(3) (2004) 421-434.
- [22] C.-M. Luo and B.-W. Cheng, Applying Even-Swap Method to Structurally Enhance the Process of Intuition Decision-Making, *Systemic Practice and Action Research* 19(1) (2006) 45-59.
- [23] D.T. Maxwell, Decision Analysis: Aiding Insight VII, *OR/MS Today* 31(5) (2004) 44-55.

- [24] J. Mustajoki and R.P. Hämmäläinen, Web-HIPRE: Global Decision Support by Value Tree and AHP analysis, *INFOR* 38(3) (2000) 208-220.
- [25] J. Mustajoki and R.P. Hämmäläinen, A Preference Programming Approach to Make the Even Swaps Method Even Easier, *Decision Analysis* 2(2) (2005) 110-123.
- [26] J. Mustajoki, R.P. Hämmäläinen and P. Lievonen, Observations from Computer-Supported Even Swaps Experiments Using the Smart-Swaps Software, *Proceedings of the TED Workshop in Human-Computer Interface Issues in e-Democracy*, Manchester, UK, November 9-11, 2005, 24-25. (Downloadable at <http://www.sal.hut.fi/Publications/pdf-files/cm05d.pdf>)
- [27] E. Natividade-Jesus, J. Coutinho-Rodrigues and C.H. Antunes, A Multicriteria Decision Support System for Housing Evaluation, *Decision Support Systems*, 43(3) (2007) 779-790.
- [28] D.J. Power and R. Sharda, Model-driven Decision Support Systems: Concepts and research directions, *Decision Support Systems* 43(3) (2007) 1044-1061.
- [29] A. Punkka and A. Salo, Preference Programming with Incomplete Ordinal Information, Manuscript, 2004. (Downloadable at <http://www.sal.hut.fi/Publications/pdf-files/mpun04.pdf>)
- [30] A. Salo, Approximate Preferences in Hierarchical Decision Models, Licentiate Thesis, Systems Analysis Laboratory, Helsinki University of Technology, 1990.
- [31] A. Salo and R.P. Hämmäläinen, Preference Assessment by Imprecise Ratio Statements, *Operations Research* 40(6) (1992) 1053-1061.
- [32] A. Salo and R.P. Hämmäläinen, Preference Programming, Manuscript, 2004. (Downloadable at <http://www.sal.hut.fi/Publications/pdf-files/msal03b.pdf>)
- [33] J.P. Shim, M. Warkentin, J.F. Courtney, D.J. Power, R. Sharda and C. Carlsson, Past, Present, and Future of Decision Support Technology, *Decision Support Systems* 33 (2002) 111-126.
- [34] E. Turban, J.E. Aronson and T.-P. Liang, *Decision Support Systems and Intelligent Systems*, 7th Edition, Prentice Hall Inc., New Jersey, 2004.

- [35] M.N. Wakshull, Application of “Even Swaps” to Normalize Qualitative and Quantitative Risk Valuations, Proceedings of the Project Management Institute Annual Seminars & Symposium, San Antonio, Texas, October 3-10, 2002.
- [36] M. Weber, Decision Making with Incomplete Information, European Journal of Operational Research 28(1) (1987) 44-57.

Table 1. Support provided by the Smart-Swaps software in the different PrOACT working phases.

Figure captions

Figure 1. The user interface for the identification of the problem, objectives and alternatives.

Figure 2. The user interface for the *Consequences* phase.

Figure 3. The user interface for the Even Swaps process.

Figure 4. Consequences table after making *Commute Time* irrelevant and information what can be achieved with a suggested swap.

Figure 5. An example of the process log.

Figure 6. A dialog for making an even swap.

Figure 7. A comparison whether to remove a practically dominated alternative.

Figure 8. The help information window.

The screenshot shows a Java Applet Window titled "Smart Swaps - Office.ssf". The window has a menu bar with "File", "View", "Options", and "Help". Below the menu bar are three tabs: "Problem / Objectives / Alternatives", "Consequences", and "Tradeoffs". The "Problem / Objectives / Alternatives" tab is active.

The "Problem" section contains a "Title:" field with the text "Alan's Office Selection Problem" and a "Problem Description:" text area. The description reads: "This example problem is adapted from the book 'Smart Choices' by Hammond, Keeney and Raiffa (Harvard Business School Press, 1999). 'Alan Miller is a computer scientist who started a technical consulting practice three years ago. For the first year he worked out of his home, but with his business growing he decided to sign a two-year lease on some space in the Pierpoint Office Park. Now that lease is about to expire. He needs to decide whether to renew it or move to a new location.'"

The "Objectives" section has an "Add Objective" button and a list of objectives: "Commute Time", "Client Access", "Office Services", "Office Size", and "Monthly Cost". Below the list are buttons for "Delete Objective", "Move Up", and "Move Down".

The "Alternatives" section has an "Add Alternative" button and a list of alternatives: "Parkway", "Lombard", "Baranov", "Montana", and "Pierpoint". Below the list are buttons for "Delete Alternative", "Move Up", and "Move Down".

The bottom of the window displays "Java Applet Window".

Figure 1. The user interface for the identification of the problem, objectives and alternatives.

	Commute Time	Client Access	Office Services	Office Size	Monthly Cost
Scale	Continuous	Continuous	A-C	Continuous	Continuous
Min/Max	Minimize	Maximize		Maximize	Minimize
Unit	min	%		sq. ft.	\$
Parkway	45	50	A	800	1850
Lombard	25	80	B	700	1700
Baranov	20	70	C	500	1500
Montana	25	85	A	950	1900
Pierpoint	30	75	C	700	1750

Figure 2. The user interface for the *Consequences* phase.

Figure 3. The user interface for the Even Swaps process.

Figure 4. Consequences table after making *Commute Time* irrelevant and information what can be achieved with a suggested swap.

Process History

Initial consequences-table

	Parkway	Lombard	Baranov	Montana	Pierpoint
Commute Time	45	25	20	25	30
Client Access	50	80	70	85	75
Office Services	A	B	C	A	C
Office Size	800	700	500	950	700
Monthly Cost	1850	1700	1500	1900	1750

* Pierpoint dominated by Lombard -> Pierpoint removed

* Parkway practically dominated by Montana -> Parkway removed

* Even Swap: Change 20 -> 25 in Baranov's Commute Time was compensated by change 70 -> 78 in Client Access.

Reduced consequences-table

	Lombard	Baranov	Montana
Commute Time	25	25	25
Client Access	80	78	85
Office Services	B	C	A
Office Size	700	500	950
Monthly Cost	1700	1500	1900

* Even Swap: Change 950 -> 700 in Montana's Office Size was compensated by change 1900 -> 1650 in Monthly Cost.

Reduced consequences-table

	Lombard	Baranov	Montana
Commute Time	25	25	25
Client Access	80	78	85
Office Services	B	C	A
Office Size	700	500	700
Monthly Cost	1700	1500	1650

Close

Java Applet Window

Figure 5. An example of the process log.

Figure 6. A dialog for making an even swap.

Remove PRACTICALLY dominated Alternative

	Montana	Parkway
Commute Time	25	45
Client Access	85	50
Office Services	A	A
Office Size	950	800
Monthly Cost	1900	1850

If the disadvantage of Montana in Monthly Cost can be compensated with its advantage in the other objectives then Parkway is practically dominated by Montana and can be eliminated.

Would you like to eliminate Parkway?

Yes No

Java Applet Window

Figure 7. A comparison whether to remove a practically dominated alternative.

Figure 8. The help information window.