An application of ranking methods: retrieving the importance order of decision factors

Anna Lázár[1] and András Kocsor[2,*]

[1]Pázmány Péter Catholic University – Information Technology Department
H-1083 Budapest, Práter utca 50/a

[2]Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged
H-6720 Szeged, Aradi vértanúk tere 1., Hungary

***Abstract* – In this paper we describe a method that returns the order of decision factors using ranking information, which can thus be interpreted as an inverse approach of the well-known Analytic Hierarchy Process (AHP). The adoption of the algorithm is conceivable in several fields, especially in those that examine or employ human decision attitudes. The accuracy of the method is investigated using both artificial and real data. In the first case we could reproduce a set of artificially generated importance orders of a fixed number of decision factors with a ninety percent correspondence, while in the second case we demonstrated how the method works when people were asked to rank 100 different sports.**

***Keywords* – Analytic Hierarchy Process, attitudes, decision theory, decision factors, ranking methodologies, ranking algorithms**

I. MOTIVATION

Methods that are well-known in the area of machine learning [3, 10, 16-18] could be a great help in psychologists practice with a few minor modifications: eliciting peoples preferences can be very helpful in almost any field. For example a well- known situation is when young adults have to decide about their goals in life: what profession should they choose? While most of them usually have more or less clear impressions about the various professions (in other words they can say how much they like or dislike them - in technical terminology they rank the professions), those are only the most unique cases when a young adult can correctly define how important a feature of a profession is when he/she has to form an opinion about it. Thus it would be most advantageous if we could somehow generate the importance of the personal decision factors from ranking information given by them. Or we can think about more abstract fields: why does someone prefer one political candidate or film star to another? Does it largely depend on his personality, appearance or talents? How big is the correlation between these aspects among the every-day people? In this paper the expression “decision factors" is a central notion which we interpret as the various aspects that play a conscious or unconscious role in forming an attitude decision on about object. The primary purpose here is to introduce a method that returns the importance order of distinct decision factors by taking ranking information as a basis.

II. INTRODUCTION

While the Analytic Hierarchy Process (AHP) [12] is a well-tried and tested method, the "reverse approach" has had much less notice as it deserves. This means that transforming between decision factor weights and ranking information is possible in either direction: from weights into ranking (which is the conventional AHP approach), and also from ranking information into decision factor weights (this is what we call the "reverse approach" a bit imprecise: it is not *exactly* the inverse of the AHP in matemathical sense, but rather in direction and approach.).The method proposed in this paper returns weights from ranking information and we also apply these values in the experiments for creating ranking values again with the intention of comparing real rankings with rankings obtained from other methods. Here it should be mentioned that we only recommend the present process be used for retrieving the importance *order* of the decision factors, not merely the weights. Several reasons suggest this decision. Firstly, numerous studies [1, 4, 7, 11, 15] argue that weights obtained from multi-attribute value trees are not too precise: the exact value depends on the structure of the tree, even when we employ the same methodology. Secondly, the order of the factor weights are more stable than their mere value. The paper is structured as follows: The next section gives a brief overview of the AHP method. Section 4 describes the decision factorization (DF) method - the reverse approach of AHP -, while Section 5 provides an account of the experiments done using both artificial and real data. Finally, we round off the paper with concluding remarks and suggestions for future study.

[*] The author was supported by the János Bolyai fellowship of the Hungarian Academy of Sciences.

III. THE AHP METHOD

Here we only undertake to briefly summarize the AHP method without the requirements of completeness. For further details the interested reader may consult [13, 14] if they wish. The first step in the AHP method is to divide the problem into sub-problems, which are structured into hierarchical levels. The number of levels depends on the complexity of the initial problem. The leaves contain the possible choices. The next step is to establish the pairwise comparison matrices for each level. These are used both for weighting the factors and weighting the possible choices from each viewpoint, one after the other. In plain terms, decision makers are asked to assign an importance weight from a scale of 1, 3, 5, 7 and 9, from "of equal importance" to "extreme importance". In some cases the intermediate values of 2, 4, 6, 8 can also be used. Since the a_{ij}-th element of the pairwise comparison matrix shows how many times the i-th element is more important than the j-th element, the a_{ji}-th element will be its reciprocal. These matrices are positive and reciprocal matrices, i.e. $a_{ij} > 0$ and $a_{ij} = 1/a_{ji}$ for $\forall\ i, j = 1, \ldots, n$.

For each viewpoint, the experts have to perform (n/2)(n-1) comparisons for a category of n elements. Once these pairwise comparison matrices are filled, the corresponding weights for the elements can be found by solving the appropriate eigenvalue-eigenvector equation. The desired weights are identified by examining the eigenvector associated with the largest eigenvalue. Once these importance-weights of the various decision factors are known, the ranking information can easily be evaluated: First we have an object (with known factor-values) which we want to rank, and second we also know the personal importance of these factors (these are the weights): the rank of the object is defined by the weighted sum of the proper factor values.

IV. DECISION FACTORIZATION: THE REVERSE APPROACH OF THE AHP METHOD

This time we take the ranking-values offered by the decision maker as a starting-point, and we attempt to produce the weights of the decision factors that appear in the ranking-decisions. For this reason we make the following assumptions:

i) It makes sense to talk about `collective weights' associated with a set of decision factors, and it is possible to express them in terms of the particular weights.

ii) Ranking algorithms are available (cf. [6]). Now we will furnish the skeleton that describes the ranking algorithms in a nutshell. Suppose we have a sequence of instances (represented by n dimensional vectors) and a rank for each of them. This is formally a series of $(\mathbf{x}_1, r_1), \ldots, (\mathbf{x}_k;\ r_k)$, where $\mathbf{x}_i \in \Re^n$ and $r_i \in \{1, \ldots, z\}$ is a finite set of integers. Each $\mathbf{x}_i$ vector represents an object described with n features, while r_i is the rank of the ith instance. Without loss of generality we may assume that the $\{1, \ldots, z\}$ finite set is ordered with the natural "<" relation. That means that the instance x_i is better than x_j, if $y_i > y_j$. The aim of a ranking algorithm is to learn a person's taste, who ranked the instance-list: after t rounds, getting a new x_{t+1} instance to predict its r_{t+1} rank as properly as possible. The general make-up of an online ranking algorithm is:

Loop: for t = 1,2,...,length of the object-list
- predict $\hat{y}_t$, the suggested rank of the x_t object
- get y_t, the real rank of x_t and update the prediction rule
End Loop

(Although we outline here the schema of the online ranking algorithms, one may apply offline methods as well. An advantage of an online method is that it can be used in web applications with dynamically changing databases)

iii) There is an expressible relation between the 'collective weights' of a set of decision factors (feature components) and the ranking loss we obtain from a ranking algorithm applied to the same features.

Before outlining the method, we still need to give a mathematical formulation for the `ranking loss' and the `collective weight'.

Ranking loss. Let us number the decision factors from 1 to n, and define the following set of subsets:

$$S = \{X \mid X \subseteq \{1, \ldots, n\}\ \}; \qquad (1)$$

So every $S_j \in S$ is a set of decision factors. Let $Loss_{S_j}$ be the ranking loss, i.e. the difference between the true rank and the predicted rank. $Loss_{S_j}$ denotes the sum of the losses accumulated during the run, divided by the number of the rounds (- which we will denote with T):

(2)

Collective weight. Let the collective weight belonging to the factors that are in the S_j set be denoted by $C(S_j)$. Of the many possibilities the most popular knowledge source integration rules are the rule of the sum, product, maximum, minimum and median. These rules lead to the following collective weights, where w_i denotes the weight of the ith decision factor:

- The sum rule: $C(S_j) = \sum_{i \in S_j} w_i$ (3)
- The product rule: $C(S_j) = \prod_{i \in S_j} w_i$
- The maximum rule: $C(S_j) = \max_{i \in S_j} w_i$
- The minimum rule: $C(S_j) = \min_{i \in S_j} w_i$

$$C(S_j) = med_{i \in S_j} w_i$$

- The median rule:

Decision factorization method. Let us introduce an $f:\Re \rightarrow \Re$ function, that converts the ranking loss into, say, a `performance' value. f can be defined for example as $f(Losss_{S_j}) = 1/Losss_{S_j}$ or $f(Losss_{S_j}) = -\log(Losss_{S_j})$ or some other similar function. Now choose a function f, a ranking algorithm and a knowledge source integration rule for the calculation of the collective weights. With this and the previous considerations we can outline the skeleton of our decision factorization method (DF).

1) Select l pairwise, distinct decision factor sets: $S_1,\ldots,S_l$.

2) Run the ranking algorithm for all S_j employing its decision factors.

3) Measure the ranking loss $Losss_{S_j}$ for all j and compute the performance values using the f function.

4) Solve the following optimization problem

$$\hat{w} = \arg\min_w \sum_{j=1}^{l} (C(S_j) - f(Loss_{S_j}))^2 \qquad (4)$$

Return $\hat{\mathbf{w}}$, the weights of the decision factors.

(In the 4th step instead of optimizing the least square error other error functions can also be applied.)

Different assumptions about the collective weight and f function lead to an optimization task that requires using different optimization methods. In this paper we employ the approach defined by the sum rule. In this case eq. (4) takes the following form in matrix notation:

$$\hat{w} = \arg\min_w \sum_{j=1}^{l} (C(S_j) - f(Loss_{S_j}))^2 = \| Aw - F \|_2^2 \qquad (5)$$

where

$$a_{ij} = \begin{cases} 1 & if\ i \in S_j \\ 0 & if\ i \notin S_j \end{cases} \qquad (6)$$

and $F = (f(Losss_{S_1}), \ldots, f(Losss_{S_l}))^T$.

The solution is : $\hat{w} = A^+ F$

where A^+ denotes the Moore & Penrose pseudo inverse [2, 5], which always exists and can be readily computed.

V. EXPERIMENTS

In this section we first introduce three measure functions which we used for demonstrating the performance of the method. Then in Section *B* we give an account of the experiments that we performed on artificial data, while Section *C* contains our observations on real data. In the experiments Crammer and Singer's `PRank' algorithm [6] was used as a ranking algorithm and the function f was chosen to be *-log(z)*.

A. Measure functions

Kendall's τ. This is used for comparing the ordinal correlation between two sequences of numbers [9, 8]. Thus we have to define the values C and D that τ applies:

C is the number of concordant pairs, D is the number of discordant pairs, and an *i,j* pair is *concordant* if both sequences order it in the same way, and *discordant* otherwise. Then

$$\tau = \frac{C-D}{C+D} = 1 - \frac{2D}{C+D} = 1 - \frac{2D}{\binom{m}{2}}$$

where the sum of C and D gives the number of all possible pairs, 2 over m, where m is the length of sequence, i.e. the number of decision factors used in this paper. Note that the totally equal order - the identity - indicates 1, random orders give about zero, while the reverse order is represented with a -1.

The number of correct positions. This indicates that how many positions match the exact value. In this paper, how many factors have been ordered to the right position in the importance order.

Consistency. Since we approximated the ***F*** performance vector by A**w** (see eq. (5)), measuring the angle γ between ***F*** and A**w** vectors can provide some useful information:

$$\cos(\gamma) = \frac{(Aw)^T F}{\| Aw \| \| F \|}$$

B. Tests on Artificial data

In order to investigate the performance of the DF method we carried out the following experiments. First, we generated 500 7-dimensional random vectors uniformly from the unit square $[0; 1]^7$. These vectors form a 500 by 7 matrix, which may be interpreted as 500 objects, each described with 7 features. Then we randomly generated 50 preference-weights: these are feature-orders where the first place contains the most important feature, the second place the second most important factor, and so on. With these data sets we could generate the ranking-vectors (50 of them) that one would give with the preference-order we generated above. Then we added a normally distributed noise with a zero mean and a standard deviation of 0.125 to these ranking vectors. Generating ranking-vectors like these makes the calculation of Kendall's τ possible.

Table I. Results using artificial data. The rows correspond to Kendall's τ, the number of correct positions and consistency, while the columns correspond to 6 sets of decision factor subsets. Set_k has k over 7 subsets, each containing k decision factor(s). Each k value in the table should be interpreted as an average of the 50 values.

	Set_1	Set_2	Set_3	Set_4	Set_5	Set_6
τ value	0.47	0.60	0.74	0.78	0.76	0.74
τ in %	73.5%	80%	87%	89.43%	88%	87%
No of correct positions	4.22	5.24	5.82	6.24	6.06	5.96
in %	60%	74.86%	83%	89%	86.57%	85%
consistency ($\cos(\gamma)$)	1	0.9998	0.9997	0.9996	0.9995	1

We know the output of the DF algorithm, and we also know the 'real` preference-orders since we defined them before.

After doing these we formed 6 sets of feature-subsets (decision factors): Set_1 contained 1 over 7, that is 7 subsets, each with one (distinct) features in it. Set_2 contained 2 over 7 that is 21 subsets, each subset with 2 features in it,…, S_k contained k over 7 subsets, each with k elements in it ($1 \leq k \leq 6$). Then, after applying our DF method, we computed Kendall's τ, the number of correct positions and the consistency for each set of subsets. The results are given in Table 1.

In the Set_1 and Set_6 cases the consistency value was 1. This was because 1 over 7 = 6 over 7 = 7, the number of unknown values in $A\mathbf{w} = \boldsymbol{F}$ (cf. eq. (5)) and the number of equations are the same. In the other cases, when we had more equations than unknown values, the high value of γ indicated that the sum rule assumption on the collective weight led to a reliable and consistent result. The high values of τ and the number of correct positions also indicate the reliability of the method. Further, notice that the accuracy depends on the size of the feature subsets. As regards the τ values the method attains its maximum in the case of Set_4.

C. Tests on Real data

With the aim of investigating the method on real data we made a list that contained one hundred kinds of sports, and valued all of them from nine points of view. (For example how spectacular, how tiring the given sport generally is, is it a ball-game, is it aquatic sport, etc.) In mathematical terms: we replaced each sport with a nine dimensional vector, where every element of the vector represented one of the nine features. Then we asked 60 people to rank all the sports from 1 to 10: 1 if he/she didn't like it at all, and 10 if he/she was "enthusiastic" about it. We intentionally didn't let them know what we meant about "liking" a sport: we asked purely for the "individual sympathy". These ranking lists then provided the input data for the DF algorithm, which returned a first set of decision factor weights. Afterwards, we asked the same people to fill out an AHP pairwise comparison matrix as well: they were asked to compare the importance of the nine features while making a ranking decision.

Table II. Test results on real data. The numbers indicate the average ranking loss.

	PRank algorithm	AHP method	DF algorithm
Average ranking loss	2.8	3.1	2.1

These matrices - using an eigenanalysis - provided a second set of decision factor weights, those that could be obtained with the well known AHP methods.

With values for these two kinds of weights we could generate two ranking-vectors. Then we compared them with the real ranks (given "in direct") using the ranking loss function. Furthermore, as a reference point we also compared the real rank values with the rank values provided directly by the 'PRank' ranking algorithm when all the features were used. (In this case for the computation of the ranking loss we only took into account the second 50 ranking values and treated the first 50 as training data.)

As it can be seen in Table 2, the DF algorithm provided 32% more proper ranking values than the weights obtained from the AHP matrices provided, and achieved a 25% better accuracy than the PRank algorithm in an immediate use. We also evaluated the AHP matrices based on another consistency definition ($(n-\lambda_{max})/(n-1)$), which is commonly used in AHP methodology, and obtained an average value 0.096. Since this number indicates such a high consistency, we came to the conclusion that while people are quite consistent in comparing the decision factor pairs, they still cannot make their ranking decisions according to these factor weights. Overall these results correspond to the observation, frequently described by psychologists that while people think they know their own decision factor weights at least approximately, in real life they influenced by different weights.

VI. CONCLUSIONS AND FUTURE WORK

While AHP gives quite fair decision factor weights in conscious decisions, comparison matrices do not provide accurate information about less conscious decision factor weights. In spite of this, these decision factor preferences also can be attained from ranking information. Hence first of all we recommend the method be applied to cases when the required decision factor preferences are applied to less conscious decisions (see section I for motivations). The Decision Factorization (DF) method introduced in this paper first defines different sets of decision factors. Then it executes a ranking algorithm and computes the ranking loss on each of these sets. This step is followed by expressing a connection between the collective weights and ranking loss that are associated with the previous sets of decision factors. This leads to the optimization problem defined by eq. (4) which is then solved. Lastly, we are planning to test the performance of the method with different ranking algorithms and with other knowledge source integration rules as well. In another future experiment we also intend to check the DF method on web-based applications with dynamically changing databases such as online bookstores where customers can vote for the books they have read.

REFERENCES

[1] Belton, V., A "Comparison of the Analytic Hierarchy Process and a Simple Multi-Attribute Value Function", *European Journal of Operat. Res*., 26, pp. 7-21, 1986.

[2] Ben-Israel, A., Greville, T., "Generalized Inverses: Theory and Applications", *Willey-Interscience*, 1974.

[3] Bishop, C. M., *Neural Networks for Pattern Recognition*, Oxford Univ. Press, 1995.

[4] Borcherding, K., Eppel, T., von Winterfeldt, D., "Comparison of Weighting Judgments in Multiattribute Utility Measurement" *Management Science*, 37, 12, pp.1603-1619, 1991.

[5] Cambell, S., Meyer, C., "Generalized Inverses of Linear Transformations", *Pitman*, 1979.

[6] Crammer, K. and Singer, Y., "Pranking with Ranking", *Advances in Neural Information Processing Systems* , 2002

[7] Hobbs, B.F., "A Comparison of Weighting Methods in Power Plant Siting", *Decision Science*, Vol. 11, pp. 725-737, 1980.

[8] Joachims, T., "Optimizing Search Engines using Clickthrough Data", *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining* (KDD), 2002.

[9] Kendall, M., Gibbons, J. D., *Rank Correlation Methods*, Oxford Univ. Press, 5th ed., 1990.

[10] Mitchell, T. M., *Machine learning*, McGraw-Hill, 1997.

[11] Pöyhönen, M., Hmlinen, R.P., Salo, A., "An experiment on the numerical modeling of verbal ratio statements", *Journal of Multi-Criteria Decision Analysis*, Vol. 6, pp. 1-10, 1997.

[12] Saaty, T.L. *The Analytic Hierarchy Process* McGraw-Hill,New York, 1980.

[13] Saaty, T.L., *Multicriteria Decision Making: The Analytic Hierarchy Process*, RWS Publications, Pittsburgh, PA, 1990.

[14] Saaty, T.L., *Fundamentals of Decision Making and Priority Theory with the Analytic Hierarchy Process*, RWS Publications, Pittsburgh, PA, 1994.

[15] Schoemaker, P.J. and Waid, C.C., "An Experimental Comparison of Different Approaches to Determining Weights in Additive Value Models", *Management Science*, 28, 2, pp. 182-196, 1982.

[16] Vladimir N. Vapnik, *The nature of statistical learning theory*, Springer, 1995.

[17] Vladimir N. Vapnik, *Statistical learning theory*, Wiley, 1998.

[18] Widrow, B. and Hoff, M. E., *Adaptive switching circuits*, 1960 IRE WESCON Convention Record, 1960, Reprinted in *Neurocomputing* (MIT Press, 1988).