Generating Packages for Preference Elicitation in E-negotiations

Vivi Nastase* Gregory Kersten†

*School of Information Technology and Engineering, University of Ottawa
800 King Edward, Ottawa, Ontario K1N 6N5, Canada
vnastase@site.uottawa.ca

†John Molson School of Business, Concordia University
1455 Blvd. de Maisonneuve West, Montreal, Quebec, H3G 1M8, Canada
gregory.kersten@jmsb.concordia.ca

Abstract

In this paper we propose an algorithm that builds a set of packages to be rated by the user, such that this set is the smallest set that would allow a system to recompute individual option ratings from the package ratings. We look at the problem of recomputing individual option ratings from package ratings as solving a linear system of equations. We will show that for a negotiation with n distinct issues, and N individual options, the user must rate $N - n + 1$ packages, and, optionally, supply $n - 1$ individual option ratings.

1 Introduction

Negotiation support systems (NSSs) are designed to help the negotiators reach an (optimal) agreement by offering analytical support (www.smartsettle.com), communication support [Schoop et al.2003], or both [Kersten and Noronha1999]. Analytical support comes in the form of numerical evaluation of the offers exchanged, which allows the negotiators to assess their position in the negotiation.

Some negotiation support systems (NSSs) like Inspire, offer the user a variety of tools that allow the user to assess his position at the negotiation table. Most of these tools use numerical values – *utility* or *preference* values – computed based on the individual choices made by the negotiator on the issues that are under debate. Obtaining user preferences raises a number of questions:

preference elicitation: what is the best way to obtain preference values from the user, for the negotiation at hand? There are three possible approaches to elicit preferences:

- the user assigns ratings to individual issue options. The disadvantage for this approach is that it does not allow the user to capture salient combinations of issues, in cases where the combination of individual values does not reflect correctly the user's preferences.

- the user assigns ratings to packages of issue options. A **package** is a combination of options, which covers all the issues under debate (one option per issue). The downside of this approach is that the utility of packages generated on the fly in the course of a negotiation is harder to compute. There may also be cases where based on the user ratings, a value for individual options cannot be computed.
- the user assigns utility values to both individual issue options, and to packages that combine all issues under debate. In this case the user can capture interaction between issue options of interest, and it allows the system to adjust the individual option ratings based on the package ratings. The problem of recomputing the individual option ratings remains, as there may be situations in which there exists no set of ratings that can satisfy the package ratings chosen by the user.

utility function: how is the utility value of a package computed from individual option utilities? Regardless of the way in which the utility/preference values are obtained from the user, in order to be able to assist the negotiation process, the system must be able to compute the utility values of the packages exchanged during the negotiation. While multiplicative approaches were proposed [Keeney and Raiffa1976], we choose to study additive functions, which are robust [Stewart1996], and also the functions of choice in Inspire [Kersten and Noronha1999].

usability: how big is the burden of assigning preference values for individual option/packages in a NSS for a user? In situations where there are many issues under debate, or issues with numerous options, asking the user to assign many values may not be a good approach.

The approach that seems to find the best combination of answers for the three questions posed by preference elicitation, seems to be:

- use both individual option and package ratings to capture a better "image" of what the user's preferences are,
- use an additive function to compute the preference/utility value of a package based on individual values,
- reduce the number of packages that the user has to rate, in order to make the package rating task less of a burden.

Building a set of packages to be rated by the user, such that it is as small as possible, while allowing a system to recompute individual option preferences, is the focus of this paper. In order to design an algorithm that builds such a package, we will look at the problem of recomputing individual option ratings based on package ratings as a system of equations whose solution we are trying to find. The individual option ratings are linearly combined to determine the value of the package.

2 Computing individual options by solving a linear system of equations

Let us denote by I_i, $i = 1, n$ the issues under debate in a given negotiation, where n is the number of issues. The issues can be *price, delivery conditions, quantity,* etc. Each issue can have a different number of options (for example price can be any valid monetary value between 10$ and 20$). N_i , $i = 1, n$ denote the number of individual options that an issue has.

For simplicity, let us denote the possible options (without distinguishing the issue they pertain to) by O_k, $k = 1, N$, where $N = \sum_{i=1}^{n} N_i$ is the total number of options across all issues.

A package is a combination of options such that each option represents one and only one issue, and an issue is represented by only one option. For example, if the issues under debate are *quantity* and *price*, and *quantity* has 3 possible values 10,000 15,000 20,000 and price has 4 possible values 1.5$ 1.75$ 2$ 2.25$, then a package would be ⟨ `10,000;1.75$` ⟩.

If we want to represent a package using the vector of possible options, we can use the following format: ⟨ `0 ... 0 1 0 ... 0 1 ...` ⟩ where a 1 that appears in position k in this vector signifies that option O_k appears in this package. In the example above, when the issues discussed are *quantity* and *price*, each with 3 and respectively 4 options, the vector of possible options is

⟨ `1.5$ 1.75$ 2$ 2.25$ 10,000 15,000 20,000` ⟩,

and the package ⟨ `1.75$; 10,000` ⟩ will correspond to the vector:

⟨ `0 1 0 0 1 0 0` ⟩

Let us denote by x_k, $k = 1, N$ the utility value for each issue option. If the user assigned the rating 60 to the package above, we could represent it through the equation

$x_2 + x_5 = 60$,

where x_2 is the utility of option O_2 (1.75$), and x_5 is the utility of option O_5 (10,000), and the utility values of individual issues are linearly combined to determine the value of the entire package.

The equation above can be equivalently rewritten as:

$a_1x_1 + a_2x2 + ... + a_5x_5 + ... + a_7x_7 = 60$

where $a_2 = a_5 = 1; a_1 = a_3 = a_4 = a_6 = a_7 = 0$.

According to this notation, the packages that the user rates can be expressed through these equations. For a set m of packages rated by the user with values y_l, $l = 1, m$, we can write m equations:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = y_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = y_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = y_m \end{cases}$$

or equivalently,

$$a_{l1}x_1 + a_{l2}x_2 + ... + a_{ln}x_n = y_l; \quad \text{where} \quad l = 1, m$$

or

$$A \cdot x = y;$$

$$\text{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}; x = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{pmatrix}; y = \begin{pmatrix} y_1 & y_2 & \cdots & y_m \end{pmatrix}$$

and $a_{ij} \in \{0, 1\}$, $a_{ij} = 0$ means in package i, option O_j with utility x_j is not included, $a_{ij} = 1$ means in package i, option O_j with utility x_j is included.

When negotiators rate packages and not individual options, we can find the utility values for individual options by solving the system of equations above. In order to have a unique solution for the system of equations, m must be equal to n, and $det(A) \neq 0$.

3 Finding a combination of packages

The problem we wish to solve is finding the smallest number of packages that a user needs to rate, in order for us to be able to compute the individual option ratings. This would allow us to compute the value of the packages the negotiators exchange during the negotiation, and plot them as a visual aid of the negotiation process for the users. Finding the packages to be rated by the users is equivalent to determining matrix A – finding the combinations of 0 or 1 values for coefficients a_{ij} such that each line in the matrix will represent a package, and such that $det(A) \neq 0$, which will allow us to compute vector x.

The constraint that each line of matrix A should represent a package, will give us an interesting property for this matrix. If there are n issues under discussion in a negotiation, each with N_i possible values ($i = 1, n$), then matrix A will have the following property:

On every line l of the matrix A, for each issue under discussion, at most one of the N_i elements of the issue I_i can have a coefficient with value 1, the rest must be 0.

$$\underbrace{a_{l1}\cdots a_{lN_1}}_{\text{coef. for issue 1}}\ \Big|\ \underbrace{a_{lN_1+1}\cdots a_{lN_1+N_2}}_{\text{coef. for issue 2}}\Big|\cdots\Big|\ \underbrace{a_{lN_1+N_2+\ldots+N_{n-1}}\cdots a_{lN_1+N_2+\ldots+N_{n-1}+N_n}}_{\text{coef. for issue n}}$$

$\sum_{i=1}^{n} a_{l\sum_{j=1}^{i} N_j} \leq 1;$

We have established then that in order for us to be able to compute individual option ratings for N options (N_i options for $i = 1, n$ issues), we need to have an NxN matrix, with a nonzero determinant, whose elements can be 0 or 1 (indicating than an issue is present or not in a specific package). Each line in the matrix will then correspond to a package. We have also established that in order for the matrix to represent possible packages, out of N_i coefficients corresponding to each option of an issue I_i, at most one can be 1, the rest must be 0 (we cannot have two prices in one package, for example, or more than one delivery date). This property will separate the matrix A into several regions, each region will correspond to an issue [1]. In the matrix below, the columns 1 through 4 correspond to issue 1, and columns 5 through 7 correspond to issue 2. Each line corresponds to a package, as described before.

$$A = \left(\begin{array}{cccc|ccc}
a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & a_{17} \\
a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & a_{27} \\
a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\
a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\
\hline
a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} & a_{57} \\
a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} & a_{67} \\
a_{71} & a_{72} & a_{73} & a_{74} & a_{75} & a_{76} & a_{77}
\end{array}\right)$$

Any matrix has the property that changing its lines or columns through linear combination among its lines or columns does not change the determinant. This property will help us prove that we cannot build an NxN matrix, where every line is a package.

Let us assume that we have a matrix A of size NxN, in which every line represents a package. According to the property described above, columns 1 through N_1 (corresponding to options of issue 1) will all have exactly one element with value 1 on each line. The same is true for the columns corresponding to every issue.

$$A = \left(\begin{array}{cccc|ccc}
1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 \\
\hline
0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right) N = 7; n = 2;$$

[1] For clarity purposes, the examples will show a 7x7 matrix, corresponding to the 2-issue case presented in section 2. The formulae however will be given for the general case.

Let us change column 1, by giving it the sum value of column 1 through N_1, and we do the same for each first column from those corresponding to each issue I_i. All these columns will be equal, since all their elements will be 1. This means the determinant of A will be 0.

In our example, we add columns 2 through 4 to column 1, and columns 6 and 7 to column 5:

$$A = \left(\begin{array}{cccc|ccc} \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ \mathbf{1} & 1 & 0 & 0 & \mathbf{1} & 1 & 0 \\ \mathbf{1} & 0 & 1 & 0 & \mathbf{1} & 1 & 0 \\ \mathbf{1} & 0 & 0 & 1 & \mathbf{1} & 0 & 1 \\ \hline \mathbf{1} & 1 & 0 & 0 & \mathbf{1} & 0 & 0 \\ \mathbf{1} & 0 & 1 & 0 & \mathbf{1} & 1 & 0 \\ \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 1 \end{array} \right) N = 7; n = 2;$$

Users rate either complete packages (with one option per issue), or individual issues. According to the results above that shows we cannot have N rated packages (where N is the total number of individual options, over all the issues), it means that within matrix A we will have some individual issues. This translates into lines of matrix A that have only one value 1, the rest 0. Since we want to fix as few individual options as possible, to allow the user to rate packages and think globally about them, we will try to reduce as much as possible the number of lines in matrix A which have only one value 1. If we have fewer than $n-1$ such lines (where n is the number of issues in the negotiation), the determinant of the matrix will again be 0: if we have at most $n-2$ lines that represent individual option ratings, it means that at least two issues will have no individual option ratings. For simplicity, let us consider that issues I_1 and I_2 have no individual option ratings. This means that for the last $n-2$ lines of the matrix, for issues I_1 and I_2 we have all elements zero. For the first $N-(n-2)$ lines, for each issue we have exactly one element 1 per line, the rest 0. We find again the situation described above, when by adding all columns for each issue, we obtain columns with all elements 1, therefore the matrix is singular (the determinant is 0).

Therefore, the smallest number of individual options that we must rate separately is $n-1$.

So what we want to do, is generate a square matrix A, of dimensions NxN, corresponding to n issues, where $N-(n-1)$ lines correspond to packages, and $n-1$ lines correspond to individual options. The matrix must not be singular.

The matrix about which we know for sure it is not singular is the identity matrix:

$$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \multicolumn{4}{c}{\dots\dots\dots\dots} \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

We also know that replacing lines/columns of a matrix by linear combinations of lines/columns will not change the value of its determinant. We will therefore start from the identity matrix, and we will generate packages by linear combinations of lines, until the matrix will have $N-(n-1)$ packages, and $n-1$ individual options.

Package generation algorithm

1. Initialize A to the identity matrix;

$$A = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

2. We change the first N_1 lines into packages. For issue I_1, the package is already instantiated – there is one and only one element with value 1 on each line. We fill in the values for the rest of the issues, by changing (randomly, for example) exactly one element to 1, for each issue, on each of the N_1 lines.

$$A = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

3. For each following group of N_i lines, $i = 2, n$, and for $N_i - 1$ lines from each group:

   (a) Let l be the index of the current line to be processed.

   Pick a line (with index) k such that $k < l$. Because $k < l$, line $\vec{k}$[2] was already processed and represents a package. Line $\vec{k}$ should also meet the following condition: if $\vec{l}$ corresponds to group N_i, line $\vec{k}$ must not have an element with value 1 on the same column as line $\vec{l}$, but an element with value 1 on the same column as another line, with index l_1, which has not been processed yet, and has only one element equal to 1, and the rest 0.

   (b) In order to obtain a package on line $\vec{l}$, we change line $\vec{l}$ to :

   $$\vec{l} \leftarrow \vec{l} + \vec{k} - \vec{l_1}$$

   Current line: $\vec{l}$ with index 5, in group N_2; Pick line $\vec{k}$ with index 4 (has a 1 on column 6, the same as line $\vec{l_1}$ also in group N_2, which is not yet a package. Add line $\vec{k}$ (4) to line $\vec{l}$ (5).

[2]The notation $\vec{line}$ stands in for the vector representing line $line$.

$$A = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

(c) Subtract line $\vec{p}$ from line $\vec{l}$;

$$A = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & \mathbf{0} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

Since we started from the identity matrix, and we changed lines only by linear combination over the lines in the matrix, $det(A) = 1$.

For our particular example, after iterating over step 3 for $N_2 - 1$ (3-1=2) times, we obtain the final result:

$$A = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right)$$

The set of packages corresponding to this matrix is:

$\langle$ 1 0 0 0 1 0 0 $\rangle \Leftrightarrow \langle$ 1.5$; 10,000 $\rangle$

$\langle$ 0 1 0 0 0 1 0 $\rangle \Leftrightarrow \langle$ 1.75$; 15,000 $\rangle$

$\langle$ 0 0 1 0 0 0 1 $\rangle \Leftrightarrow \langle$ 2$; 20,000 $\rangle$

$\langle$ 0 0 0 1 0 1 0 $\rangle \Leftrightarrow \langle$ 2.25$; 15,000 $\rangle$

$\langle$ 0 0 0 1 1 0 0 $\rangle \Leftrightarrow \langle$ 2.25$; 10,000 $\rangle$

$\langle$ 0 0 1 0 0 1 0 $\rangle \Leftrightarrow \langle$ 2$; 15,000$\rangle$

$\langle$ 0 0 0 0 0 0 1 $\rangle \Leftrightarrow \langle$ 20,000 $\rangle$

Because for each group N_i, $i = 2, n$ we change $N_i - 1$ lines into packages, we will remain with $n - 1$ lines that correspond to single issue options.

Within the process of generating this matrix there are sequences that can be adjusted. First of all, the individual options that will be rated separately can be chosen such that these are the options with either a minimum or maximum value for a user, among the issues they belong to. Also, choosing a line (package) to combine with another line to generate a new package can be chosen such that the end result is a balanced set, in which individual options within an issue appears approximately the same number of times. Or either choice process can be random.

The number of packages to be rated by the user can range between 0 and $N - (n - 1)$, as the process of generating packages using the algorithm described can be stopped at any time. The number of packages to be generated can be a parameter for the algorithm.

4 Conclusion

We have presented an algorithm that generates packages to be rated by the user for an electronic negotiation support system that provides analytical support. The algorithm is implemented and incorporated in an NSS platform. The number of packages generated can be adjusted between 0 (when only single issue options are rated), and a maximum number: $N - (n - 1)$, where N is the total number of individual options, and n is the number of issues. These package ratings together with the individual ratings allows the system to recompute all individual option ratings. Having the individual option ratings is necessary for the NSS to compute utility values of the offers exchanged during negotiations, in order to offer analytic support to the users of the electronic negotiation support system.

References

[Keeney and Raiffa1976] R.L. Keeney and H. Raiffa. 1976. *Decisions with multiple objectives: Preferences and value tradeoffs.* J. Wiley and Sons, New York.

[Kersten and Noronha1999] Gregory Kersten and Sunil Noronha. 1999. Www-based negotiation support: Design, implementation and use. *Decision Support Systems*, 2(25):135–154.

[Schoop et al.2003] Mareike Schoop, A. Jertila, and T. List. 2003. Negoisst: A negotiation support system for electronic business-to-business negotiations in e-commerce. *Data and Knowledge Engineering*, 3(47):371–401.

[Stewart1996] T. Stewart. 1996. Robustness of additive value function methods in mcdm. *Journal of Multi-Criteria Decision Analysis*, 4(5):301–309.