

# Learning the user's preferences for multiple criteria ranking

David Isern, Aïda Valls, Antonio Moreno

Universitat Rovira i Virgili

Department of Computer Science and Mathematics

Research Group on Artificial Intelligence, BANZAI

{david.isern,aida.valls,antonio.moreno}@urv.cat

## Abstract

Multicriteria decision ranking methods (MCDM) are used to aggregate information from different preference (or utility) functions to obtain a global ranking of a given set of alternatives. However, in some domains the preferences used may change dynamically over time. In this paper, we present a learning method that adapts the user's preference functions by means of the observation of which is the alternative that the user selects from the ranking that the MCDM method has proposed.

**Keywords:** Aggregation operators, preferences, learning, multicriteria decision making.

## 1 Introduction

In traditional multicriteria decision making (MCDM) approaches the main goal of the method is to select the best alternative, or to rank or classify a set of given alternatives. Those alternatives are described by means of a set of preference attributes per each alternative. In utility-based approaches those criteria indicate the degree of satisfaction of each alternative with respect to one of the attributes that the user wants to consider. In this sense, different algorithms were developed according to the nature of the data (*i.e.* numerical or linguistic) and the weighting factor applied to the attributes and/or to the values [1, 2, 3]. Unfortunately, these methods are not designed to be used in an *on line* system where the preferences considered can change dynamically according to some criteria. In those systems, the user's preferences are modeled and stored in his personal profile to be used by decision making tools. In our approach, the algorithm proposed allows to deal with dynamic preferences by changing the user's profile values.

The process described in this paper is divided in two main tasks: first, it evaluates a set of proposals received from a recommender system, and second, it learns from actions made by the user (feedback). Relating to the first task, a rating and ranking of the proposals received is

required. This process could be performed using several well-known multicriteria decision making operators [4] or applying case-based reasoning from a set of well known past examples [5]. The next task is to adapt the user's preferences automatically by a learning process. That process could allow the system to evolve from a set of predefined preferences to the user's-centered interests.

Moreover, with this approach we know the preferences of all users and we can extract information of their behavior, for instance, know the most selected value of an attribute.

Each time that the system ranks a set of alternatives using a multicriteria decision making method, the system observes which is the alternative selected by the user, assuming that the multicriteria decision method is reliable; if the user does not choose the first one, it indicates that the preferences are not appropriately modeled in the user's profile. This is a valuable information because we set up the ranking process with different parameters and an aggregation-based operator has been designed in order to adapt the profile to that selection [6]. Therefore, we use the information of the selected alternative, and of all the discarded ones (the ones that were ranked better than the selected one) to deduce which should be the correct user's profile, that is, which are the current user's preferences.

Although we will explain this approach using a specific MCDM method, namely a LOWA operator, we believe that the ideas presented here could be used in a more general framework.

The rest of the paper is organised as follows. First of all, an analysis of the related work in the area is made. In §3 the multicriteria decision ranking process is described. Then, in §4 the learning process and dynamic adaptation of the user's profile is explained. After that, an example of the use of this method in a healthcare domain is shown. Finally, some conclusions are given.

## 2 Related work

The paper is focused in the learning preferences process applied in a multicriteria decision making application. Preference learning is recently beginning to be taken into consideration. Related works in this area such as [7, 8] learn

global preferences as an iteration of *pairwise preferences*. The goal is to use pairwise preferences from training examples for predicting a total order function that allows to rank all possible labels for a new training example. However, as pointed out in [7], this approach requires a great number of examples in the training set in order to use SVM, that sometimes could be hard to obtain. A more general framework is described in [9] where the authors learn both the attributes considered in the user’s profile and the preferred values also using SVM as a classifier to distinguish the good and bad alternatives.

Another approach is to consider the preferences as a Constraint Satisfaction Problem (CSP). In [10], the authors develop a learning method based on multiclass label, which builds a constraint classifier between each example and the rest (is a case of ”one versus all” approach). This approach allows to determine a partial order of that example in comparison with the rest.

In contrast, our approach is different because we do not use past examples. We learn from the most recent experience in a system that is continuously used by the user. Although we miss the information about past episodes, this model allows to react easily to new trends and new situations. Moreover, the complexity of our method is lower than that of the SVM-based approaches.

### 3 Multicriteria alternatives ranking

Our goal is to employ the user’s preferences to rate, rank and filter a set of alternatives. As it has been introduced, we are interested in a healthcare environment, where the alternatives are a set of possible appointments for a medical test, described using different criteria, such as the distance to the medical centre or the day and time of the test, among others. In this context, the user’s preference with respect to the distance/day/time may change over time. So, we propose to adapt the user profile each time that the patient uses the system to obtain a ranking of appointment proposals. Fig. 1 shows the two steps: the ranking of the alternatives and the adaptation of the user’s profile. In this section, we explain how the user’s preferences are modelled and aggregated.

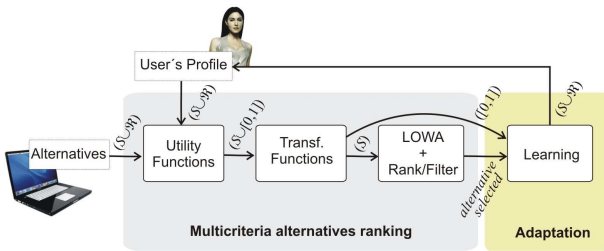


Figure 1: Decision and adaptation

#### 3.1 User’s profile

The user’s profile is a set of both linguistic and numerical variables. The scale of the linguistic variables is  $S$  and the

domain of the numerical ones is  $[0, 1]$ .

We will denote  $S = \{s_i\}$ , and  $i \in \{0, \dots, T\}$  a finite ordered set of  $T + 1$  linguistic labels whose semantics is given by fuzzy sets. Each label  $s_i$  is defined by a 4-tuple  $(x_0, x_1, x_2, x_3)$ , where  $x_1$  and  $x_2$  indicate the interval in which the membership function value is 1, and  $x_0$  and  $x_3$  are the bounds of the definition of a trapezoidal fuzzy membership function (see Fig. 3). Then, in the user’s profile we have a utility function  $U_{atr_i}^L$  that associates each possible value of the categorical attribute  $atr_i$  to a label in  $S$ , indicating its preference score. For numerical attributes, we have a utility function  $U_{atr_i}^N$  that receives the numerical value  $r$  of the corresponding attribute, and compares  $r$  with the preferred value of the user  $r_{user_i}$ . The utility function of the  $i^{th}$  attribute takes  $k_i \approx \frac{10}{(max_{atr_i} - min_{atr_i})}$ .

$$U_{atr_i}^N : \mathbb{R} \rightarrow [0, 1] \quad U_{atr_j}^L : String \rightarrow S$$

$$r \rightarrow 1/e^{k_i|r_{user_i} - r|} \quad str \rightarrow s_i$$

#### 3.2 Rating of alternatives

To rank the set of alternatives we use a classical multicriteria decision-making process with two stages: rating and ranking [4]. Being an alternative a tuple of the form  $p_i = (x_0, x_1, x_2, \dots, x_K)$  its rating is done in three steps (Fig. 1):

*Step 1)* All the values describing an alternative,  $p_i$ , are transformed into preference values in the domain  $([0, 1] \cup S)$  by applying the appropriate utility functions  $U_{atr_h}$  as described above.

*Step 2)* The numerical preferences in  $[0, 1]$  are transformed into the linguistic domain  $S$  by means of a particular numerical-linguistic transformation function defined in [1] (linguistic preferences are left without changes), obtaining the transformed vector called  $alt_i = \{a_k\}$  ( $a_k \in S$ ).

*Step 3)* The linguistic preferences in  $alt_i$  are aggregated using the LOWA operator, obtaining a linguistic rating.

Finally, all alternatives can be ranked using the rating values. Then, a filtering is performed to show to the user only the best alternatives, so that he can select one of them.

The problem of aggregating information has been widely studied [11]. There exist several methods to aggregate numerical values as well as linguistic terms. The family of OWA operators are in the class of mean operators. They are idempotent, monotonic and commutative.

The LOWA aggregation operator  $\phi$  was defined in [12]. It is an extension of the OWA operator to deal with linguistic variables. The operator  $\phi$  aggregates a set of labels  $A = \{a_1, \dots, a_m\}$ , where  $a_i \in S$ , with respect to a set of weights  $W = \{w_1, \dots, w_m\}$  such that  $w_i \in [0, 1]$  and  $\sum_i w_i = 1$ .

Those weights specify the decision-maker policy.

$$\begin{aligned}\phi(a_1, \dots, a_m) &= W \cdot B^T = C^m \{w_k, b_k, k = 1, \dots, m\} \\ &= w_1 \odot b_1 \oplus (1 - w_1) \odot C^{m-1} \{\beta_h, b_h, h = 2, \dots, m\}\end{aligned}$$

where  $\beta_h = w_h / \sum_2^m w_h, h = \{2, \dots, m\}$  and  $B = \{b_1, \dots, b_m\}$  is a permutation of the elements of  $A$ , such that  $B = \sigma(A) = \{a_{\sigma(1)}, \dots, a_{\sigma(m)}\}$ , where  $a_{\sigma(j)} \leq a_{\sigma(i)} \forall i \leq j$ .  $C^m$  is the convex combination operator of  $m$  labels; if  $m = 2$ , then  $C^2\{w_i, b_i, i = 1, 2\} = w_1 \odot s_j \oplus (1 - w_1) \odot s_i = s_k, s_i, s_j \in S, (i \leq j)$  such that,  $k = \min\{T, i + \text{round}(w_i \cdot (j - i))\}$ . If  $w_j = 1$  and  $w_i = 0$  with  $i \neq j$ , then  $C^m\{w_i, b_i, i = 1, m\} = b_j$

The weight vector  $w$  permits to adjust the degree of conjunction and disjunction implicit in any aggregation. This is done by using linguistic quantifiers (expressed as a set of weights) that permit to define different aggregation policies. In [2] different fuzzy majority-based policies are identified, such as “most”, “at least half” or “as many as possible”.

In a previous work ([6]) we compared the effect of using different policies in the rating results following this method. In addition, we studied the behaviour of different linguistic domains ( $S$ ) (symmetric versus non-symmetric linguistic domains were compared).

## 4 Learning Preferences

As a result of the aggregation process, the alternatives are rated with a linguistic label in  $S$ , and then ranked. If the user does not choose the first alternative, we adapt the user’s profile so that if we repeated the rating and ranking with the same alternatives and the new profile, the selected alternative would receive a better rating (if possible, a rating that would place it in the first position). If the selected alternative is the first, it is not necessary to adapt the profile.

Our method is based in the following statement: if the user has selected  $p_i$  with a rate  $a_r$  ( $a_r \in S$ ), we can use the subset of alternatives  $\{p_h, h = 0, i - 1\}$  that received better ratings (and, therefore, have better positions in the ranking) to learn which should be the user’s profile.

### 4.1 Learning algorithm

As we have just said, we will only consider the  $p_h, h \in (0 \dots i - 1)$  best ranked alternatives to learn. Having that each alternative  $p_j$  has  $K$  values for the criteria, we propose the following algorithm to adapt the user’s profile after his/her selection:

*Step 1)* All the linguistic preference values of the alternatives from position  $j = 0$  to  $j = i$  are translated to the numerical domain  $[0, 1]$  by means of a particular linguistic-numerical transformation function defined in [1] (numerical preferences are left without changes), obtaining the transformed vector called  $v_j = \{n_{j_k}\}$  ( $n_{j_k} \in [0, 1]$ ) (See Fig. 2a).

*Step 2)* Using any unsupervised clustering method, generate  $c$  clusters from the set of alternatives  $V = \{v_h, h = 0, i - 1\}$ . Then, we calculate the prototype of each cluster,  $R_j, j = 0, \dots, c - 1$  (See Fig. 2b).

*Step 3)* Find the distance between  $v_i$  (the alternative selected by the user) and all the prototypes  $R_j$ . Let  $R_{min}$  be the closest prototype to  $v_i$ , so that its distance<sup>1</sup> is  $\{\min(\text{dist}(R_j, v_i)) \forall j = 0, c - 1\}$ .

*Step 4)* Let  $A$  be the vector  $v_i$ , and  $B$  be the vector  $R_{min}$ . For each of the  $K + 1$  criteria, calculate the difference of its value in  $A$  with respect to  $B$ ,  $d_j = (a_j - b_j)$  ( $j \in 0..K$ ). The variables for which the difference  $d_j$  is greater than a given threshold, are marked to be changed in the user’s profile.

*Step 5)* To update the preference values of the criteria marked in the previous step, we propose a method based on the LOWA aggregation operator described in §3.2. Being  $A$  the vector  $v_i$ ,  $B$  the vector  $R_{min}$ , and  $m$  the variable to be updated, an intermediate value  $\alpha_m$  is calculated, so that  $\alpha_m = a_m + w_m(b_m - a_m)$ , where the weight  $w_m$  is indicating the degree of change that we want to apply to the criterion  $m$ . We propose to use the difference of the actual value and the desired one, that is  $w_m = |b_m - a_m|$ .

*Step 6)* Using the value  $\alpha_m$ , adapt the user’s profile. For numerical variables, the new value of the  $m^{\text{th}}$  variable in the user’s profile is  $\beta_m = (U_m^{N-1}(\alpha_m))$ . For linguistic variables, that value  $\beta_m$  is obtained transforming the number  $\alpha_m$  into its corresponding term in  $S$ , with the same function applied in *Step 1* and defined in [1].

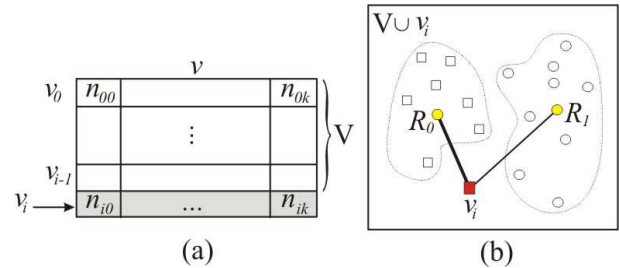


Figure 2: Clustering of alternatives in the learning process

### 4.2 Some comments on the learning method

In this section we want to present some alternatives to some of the steps of the learning method we have proposed, and argue why we have chosen this configuration and not another. Let us proceed step by step.

In step 2 we use the k-means algorithm because it is a fast and well-known method, but other non-supervised clustering techniques could be applied [13]. The number of clusters that are generated must be carefully studied. Depending on the application domain, we could consider to

<sup>1</sup>Any distance measures can be applied. We choose the Euclidean Distance

have more than 2 clusters. However, we must think that if the profile and the decision making methods are correct, the user should not select an alternative far from the initial positions of the ranking, so the number of alternatives to cluster should not be more than 10. With this assumption, building more than 3 clusters seems not necessary.

The rationale behind step 3 is that we can have sets of alternatives with common features, and we must select one of the prototypes to become our ideal alternative, that is, the one we want to get closer to. If we select the more distant prototype, we are very optimistic and want to make all the changes necessary in our preferences to arrive to the best positions. A more conservative approach is to consider the closest prototype. If we have more than 2 clusters we can select any other cluster in between.

The threshold defined in step 4 allows us to restrict the number of changes in the user's profile. Depending on the number of criteria that change, the profile is adapted more or less smoothly.

In step 5, we calculate the new value for each criterion in the profile,  $\beta_m$ . The weight  $w_m$  indicates the degree of change we will apply to the actual value in the profile. It is based on the difference between the actual value,  $a_k$  and the desired one  $b_k$ . However, other options or more parameters could be used. For example, we could allow bigger changes at the beginning of the use of the recommendation system, and after some time, take a more conservative approach, reducing the amount of change per iteration.

### 4.3 Case Study: Medical services

We have tested the proposed method for learning preferences in a multiagent system called HeCaSe2[14]. This system is intended to facilitate the provision of user-centered medical services. In particular, HeCaSe2 helps doctors to personalise medical guidelines to the treatment of each patient. During the process, if the doctor detects that some clinical test is needed, the agents are able to automatically obtain a list of possible appointments for the test. Then, using the patient preferences on different criteria, the list of appointments is ranked using the method described in §3.2. If the patient does not select the first proposal, we apply the learning process to adapt the user's profile.

VH=Very\_High (0.8125,0.8626,1.0,1.0)  
H=High (0.5125,0.5625,0.8125,0.8625)  
M=Medium (0.4375,0.4875,0.5125,0.5625)  
L=Low(0.325,0.375,0.4375,0.4875)  
QL=Quite\_Low (0.1875,0.2375,0.325,0.375)  
VL=Very\_Low (0.05,0.1,0.1875,0.2375)  
N=None (0.0,0.0,0.05,0.1)

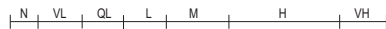


Figure 3: Linguistic domain used

In this application we consider 5 variables. Three of them are linguistic variables: *day\_of\_week*, *centre* (destination medical centre) and *period\_day* (morning, afternoon, night); and two are numerical: *distance* (kilometers from the origin centre to the destination) and *delay\_days* (days

to wait before the test). Fig. 3 shows the linguistic domain  $S$  of the variables. After the study made in [6], we have selected a vocabulary with seven linguistic labels non symmetrically distributed, to have more precision to indicate different bad degrees of preference.

For our example, let us consider that the profile of Mrs. Smith is this one:

*delay\_days* <0.0>  
*distance* <0.0>  
*centre* <<(MCBona,N)(MCBorges,M) (MCConst,H)  
(MCMorell,VH) (MCGimb,M) (MCHospi,L)  
(MCJaume,QL) (MCLlib,L)>>  
*day\_of\_week* <<(Sun,VL) (Mon,QL) (Tue,M) (Wed,H)  
(Thu,M) (Fri,M) (Sat,L)>>  
*period\_day* <<(Morning,VH) (Afternoon,QL) (Night,VL)>>

Mrs. Smith needs a tooth X-ray to check which is the origin of her recurrent toothache. Each of the appointments that the system finds is initially considered a valid alternative. Each alternative is a 5-tuple  $p_i = \langle \text{delay\_days}, \text{distance}, \text{medical\_centre}, \text{day\_of\_week}, \text{period\_day} \rangle$ . The values on each alternative are evaluated using Mrs. Smith's utility functions stored in her profile, in order to know the corresponding linguistic preference values (following [12]). Let's consider the we have found six possible appointments  $P$  for Mrs. Smith:

$p_0 : (2.0, 1.2, MCBorges, Wed, Morn) \xrightarrow[\text{transf}]{U} (H H M H VH)$   
 $p_1 : (1.0, 8.0, MCConst, Mon, Aft) \rightarrow (H VL H QL QL)$   
 $p_2 : (4.0, 9.0, MCBona, Thu, Aft) \rightarrow (L VL N M QL)$   
 $p_3 : (5.0, 1.2, MCBorges, Sat, Night) \rightarrow (QL H M L VL)$   
 $p_4 : (10.0, 1.2, MCBorges, Fri, Morn) \rightarrow (N H M M VH)$   
 $p_5 : (9.0, 17.0, MCHospi, Fri, Aft) \rightarrow (VL N L M QL)$

The next step consists of applying the LOWA operator to rate the proposals. An important parameter to be fixed in this stage is the weight vector  $W$ . As it has been mentioned before, weights specify different aggregation policies. In this application we have good results with the policy "as many as possible" [2], so weights are:  $W = (.0, .0, .2, .4, .4)$ . After applying the LOWA operator, all alternatives are linguistically rated and can be ranked and presented to the user for a selection. In the example, the ranked list of appointments for Mrs. Smith is the next one:

$p_0 : (H H M H VH) \rightarrow (H)$   
 $p_4 : (N H M M VH) \rightarrow (H)$   
 $p_1 : (H VL H QL QL) \rightarrow (M)$   
 $p_3 : (QL H M L VL) \rightarrow (M)$   
 $p_2 : (L VL N M QL) \rightarrow (L)$   
 $p_5 : (VL N L M QL) \rightarrow (L)$

Let us suppose that Mrs. Smith is not suffering from toothache at the moment, and that she needs some time to arrange her schedule to fix the medical appointment, because she must find a babysitter for her twins. So, she selects option  $p_5$ , although it is in the fifth position of the ranking. So, we assume that her profile is not correct and we will try to modify it. In fact, if we observe the profile, we will see that the most preferred option is to wait 0 days, but this is not the case of Mrs. Smith.

Now, we start the learning process to adapt the user's profile to the selection made. We build the set  $V = \{v_0, v_4, v_1, v_3, v_2\}$  and make two clusters using the  $k$ Means algorithm. To apply this method we need to have all the information in a numerical scale. To do this we apply the linguistic-numerical transformation function described in [12] (see Fig. 1). As a result of that transformation, the alternatives are described as follows<sup>2</sup>:

$$\begin{aligned}
v_0 &: (\text{H H M H VH}) \leftrightarrow \\
&\quad (0.6065 \ 0.7408 \ 0.4999 \ 0.6875 \ 0.9259) \\
v_4 &: (\text{N H M M VH}) \leftrightarrow \\
&\quad (0.0821 \ 0.7408 \ 0.4999 \ 0.4999 \ 0.9259) \\
v_1 &: (\text{H VL H QL QL}) \leftrightarrow \\
&\quad (0.7788 \ 0.1353 \ 0.6875 \ 0.2812 \ 0.2812) \\
v_3 &: (\text{QL H M L VL}) \leftrightarrow \\
&\quad (0.2865 \ 0.74082 \ 0.4999 \ 0.4062 \ 0.1437) \\
v_2 &: (\text{L VL N M QL}) \leftrightarrow \\
&\quad (0.3679 \ 0.1054 \ 0.03055 \ 0.4999 \ 0.2812) \\
v_5^* &: (\text{VL N L M QL}) \leftrightarrow \\
&\quad (0.1054 \ 0.0143 \ 0.4062 \ 0.4999 \ 0.2812)
\end{aligned}$$

The  $k$ Means algorithm generates two clusters  $\{v_1, v_2, v_3\}$  and  $\{v_0, v_4\}$ , with their corresponding centroids, called  $R_0$  and  $R_1$ , respectively.

$$\begin{aligned}
R_0 &: (0.4777 \ 0.3272 \ 0.4060 \ 0.3958 \ 0.2354) \\
R_1 &: (0.3443 \ 0.7408 \ 0.4999 \ 0.5937 \ 0.9259)
\end{aligned}$$

Now, we measure the distance between the selected alternative ( $v_5^*$ ) and the two prototypes. We apply the Euclidean Distance obtaining:

$$dist(v_5^*, R_0) = 0.22338 < dist(v_5^*, R_1) = 0.45125$$

Having a conservative approach, we decide that we should adapt the user's profile to be closer to  $R_0$ .

In the next step, we calculate the differences  $d_j$  between  $v_5^*$  and  $R_0$ , obtaining:

$$\begin{aligned}
v_5^* & \quad (0.1054 \ 0.0143 \ 0.4062 \ 0.4999 \ 0.2812) \\
R_0 & \quad (0.4777 \ 0.3272 \ 0.4060 \ 0.3958 \ 0.2354) \\
d & \quad (0.3723 \ 0.3129 \ 2.3150\text{E-}4 \ 0.1042 \ 0.0458)
\end{aligned}$$

At this point, we have to establish a threshold to choose which attributes are so distant to be considered to update the profile. If we choose a low threshold, too many attributes will be changed simultaneously and the profile modification will be bigger and more difficult to control. On the other hand, a threshold too high will not allow the system to react to the changes in the preferences of the user. Let us suppose that we take a threshold of 0.35; in that case, we only have to change the first attribute: the *delay\_days*.

The adaptation of that variable is made according to the aggregation-based updating function defined in the proposed learning algorithm. According to that,  $\alpha = 0.1054 +$

<sup>2</sup>To improve legibility, all numerical values have been rounded to four decimals. Internal operations maintain more precision.

$w(0.4777 - 0.1054)$  where  $w = 0.3723$ , resulting  $\alpha = 0.2440$ . This means that the preference of the value proposed in  $v_5$  should have been 0.2440 instead of 0.1054. Therefore, to change this preference utility function, we can only change the number of days considered to be the most preferred by the user. In this example, we initially assumed that Mrs. Smith wants to wait 0 days, but from the scenario, we have said that this is not true. Applying the inverse function to obtain the number of days from the preference value 0.2440, we found that the most preferred number of days is set to 5.3, which is a good approximation of the real desired value.

Now, we can check what happens with this new profile. So, we start again the ranking process with the same alternatives. The results obtained now are shown below. Notice that the change in the utility function of the *delay\_days* variable has affected the preference values of all the proposals.

$$\begin{aligned}
p_0 &: (\text{L H M H VH}) \rightarrow (\text{H}) \\
p_1 &: (\text{QL VL H QL QL}) \rightarrow (\text{L}) \\
p_2 &: (\text{H VL N M QL}) \rightarrow (\text{M}) \\
p_3 &: (\text{VH H M L VL}) \rightarrow (\text{H}) \\
p_4 &: (\text{QL H M M VH}) \rightarrow (\text{H}) \\
p_5 &: (\text{L N L M QL}) \rightarrow (\text{L})
\end{aligned}$$

Our goal was to increase the preference value of  $p_5$  smoothly. In fact, it can be observed that now  $p_5$  is considered of  $L$  (low) preference in its first attribute, instead of  $VL$  (very-low). Moreover, the changes suffered by the first variable have some effects in the ranking, changing the positions of some of the alternatives. Some alternatives with a delay close to the new preferred value have increased its global utility value such as  $p_3$ . However, in this particular example this change is not sufficient to change the overall ranking of the proposal selected by Mrs. Smith. Note that the method does not aim to improve automatically all variables in one step (that would be too drastic). Moreover, there are several parameters that affect the degree of change, such as the threshold taken in the adaptation or the centroid chosen to adapt the selected proposal.

## 5 Conclusions

We propose a method for learning the user's preferences that takes into account the feedback from the user when it makes a selection from a ranking recommended by a multicriteria decision making system. We have explained the case of using a ranking method based on the use of the LOWA operator. However, this learning method could be also applied with other ranking techniques.

We want to stress that the learning method is able to deal both with qualitative and numerical preferences. It is only required (i) to have a function to transform qualitative preference values into numbers and vice-versa and (ii) for numerical utility functions, to have an inverse function to obtain a single value in the original scale from any preference value. These constraints are due to the fact that we use numbers to learn, because with numbers we can work with more precision and modify the profile more gradually than using qualitative values.

In the tests that we have made using the HeCaSe2 prototype, when the user changes one of his preference values (like Mrs. Smith with the preferred number of days before making any test), the profile is correctly updated after some iterations of the learning method, which is also indicating that the learning is done smoothly, in order not to be too sensitive to occasional changes of the user's preferences.

In fact, it can be argued that it would be interesting to be able to introduce information about the context of each decision. For example, in the medical setting, Mrs. Smith can prefer to wait a week before a test to arrange her schedule; however, in an urgency situation, this criterion should not be used. Another example, could be the situation of having different preferences of medical centres for different types of tests. Context-situated decision making and learning is an interesting open issue to consider.

## 6 Acknowledgements

This work has been partially supported by the IST project K4CARE (IST-2004-026968) and the Spanish Research Network REDEMAP-II (REd Temática Nacional de Procesos de Toma de DEcisiones, Modelado y Agregación de Preferencias) (TIN2004-21700-E).

## References

- [1] M. Delgado, F. Herrera, E. Herrera-Viedma, and L. Martínez, "Combining numerical and linguistic information in group decision making," *Inf Sci*, vol. 107, pp. 177–194, 1998.
- [2] D. R. Yager, "On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decision making," *IEEE Trans Syst Man Cybern*, vol. 18, pp. 183–190, 1988.
- [3] V. Torra, "The Weighted OWA Operator," *International Journal of Intelligent Systems*, vol. 12, pp. 153–166, 1997.
- [4] J. Figueira, S. Greco, and M. Ehrgott, *Multiple Criteria Decision Analysis: State of the Art Surveys*, vol. 78 of *International Series in Operations Research and Management Science*. Springer, 2005.
- [5] A. Aamodt and E. Plaza, "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches," *Artificial Intelligence Communications*, vol. 7, pp. 39–59, 1994.
- [6] D. Isern, A. Valls, and A. Moreno, "Using Aggregation operators to personalize agent-based medical services," in *10th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, KES2006*, LNAI, (Bournemouth, UK), p. to appear, Springer Verlag, 2006.
- [7] J. Fürnkranz and E. Hüllermeier, "Pairwise Preference Learning and Ranking," in *14th European Conference on Machine Learning, ECML-03* (N. Lavrac, D. Gamberger, L. Todorovski, and H. Blockeel, eds.), vol. 2837 of *LNCS*, (Cavtat, Croatia), Springer Verlag, 2003.
- [8] M. T. Gervasio, M. D. Moffitt, M. E. Pollack, J. M. Taylor, and T. E. Uribe, "Active Preference Learning for Personalized Calendar Scheduling Assistance," in *10th international conference on Intelligent user interfaces* (J. Riedl, A. Jameson, D. Billsus, and T. Lau, eds.), (San Diego, California, USA), pp. 90–97, ACM Press, 2005.
- [9] G. González, C. Angulo, B. López, and J. L. Rosa, "Smart User Models: Modelling the Humans in Ambient Recommender Systems," in *Workshop on Decentralized, Agent Based and Social Approaches to User Modelling (DASUM 2005)* (P. Dolog and J. Vassileva, eds.), (Edinburgh, Scotland), pp. 11–20, 2005.
- [10] S. Har-Peled, D. Roth, and D. Zimak, "Constraint Classification: A New Approach to Multiclass Classification," in *13th International Conference on Algorithmic Learning Theory*, vol. 2533 of *LNCS*, (Lübeck, Germany), pp. 365–379, Springer, 2002.
- [11] M. Grabisch, S. Orlovski, and R. Yager, "Fuzzy aggregation of numerical preferences," in *Fuzzy sets in decision analysis, operations research and statistics* (R. Slowinski, ed.), pp. 31–68, Kluwer Academic, 1999.
- [12] F. Herrera and E. Herrera-Viedma, "Linguistic decision analysis: steps for solving decision problems under linguistic information," *Fuzzy Sets and Systems*, vol. 115, pp. 67–82, 2000.
- [13] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann, 2nd edition ed., 2005.
- [14] D. Isern and A. Moreno, "Agent-Based Careflow Usign CPGs," in *Recent Advances in Artificial Intelligence Research and Development* (J. Vitrià, P. Radeva, and I. Aguiló, eds.), vol. 113 of *Frontiers in Artificial Intelligence and Applications*, (Barcelona, Catalonia), pp. 11–18, IOS Press, 2004.