



DYNAMIC ADAPTATION OF USER PROFILES IN RECOMMENDER SYSTEMS

Lucas Marín Isern

Dipòsit Legal: T.1292-2013

ADVERTIMENT. L'accés als continguts d'aquesta tesi doctoral i la seva utilització ha de respectar els drets de la persona autora. Pot ser utilitzada per a consulta o estudi personal, així com en activitats o materials d'investigació i docència en els termes establerts a l'art. 32 del Text Refós de la Llei de Propietat Intel·lectual (RDL 1/1996). Per altres utilitzacions es requereix l'autorització prèvia i expressa de la persona autora. En qualsevol cas, en la utilització dels seus continguts caldrà indicar de forma clara el nom i cognoms de la persona autora i el títol de la tesi doctoral. No s'autoritza la seva reproducció o altres formes d'explotació efectuades amb finalitats de lucre ni la seva comunicació pública des d'un lloc aliè al servei TDX. Tampoc s'autoritza la presentació del seu contingut en una finestra o marc aliè a TDX (framing). Aquesta reserva de drets afecta tant als continguts de la tesi com als seus resums i índexs.

ADVERTENCIA. El acceso a los contenidos de esta tesis doctoral y su utilización debe respetar los derechos de la persona autora. Puede ser utilizada para consulta o estudio personal, así como en actividades o materiales de investigación y docencia en los términos establecidos en el art. 32 del Texto Refundido de la Ley de Propiedad Intelectual (RDL 1/1996). Para otros usos se requiere la autorización previa y expresa de la persona autora. En cualquier caso, en la utilización de sus contenidos se deberá indicar de forma clara el nombre y apellidos de la persona autora y el título de la tesis doctoral. No se autoriza su reproducción u otras formas de explotación efectuadas con fines lucrativos ni su comunicación pública desde un sitio ajeno al servicio TDR. Tampoco se autoriza la presentación de su contenido en una ventana o marco ajeno a TDR (framing). Esta reserva de derechos afecta tanto al contenido de la tesis como a sus resúmenes e índices.

WARNING. Access to the contents of this doctoral thesis and its use must respect the rights of the author. It can be used for reference or private study, as well as research and learning activities or materials in the terms established by the 32nd article of the Spanish Consolidated Copyright Act (RDL 1/1996). Express and previous authorization of the author is required for any other uses. In any case, when using its content, full name of the author and title of the thesis must be clearly indicated. Reproduction or other forms of for profit use or public communication from outside TDX service is not allowed. Presentation of its content in a window or frame external to TDX (framing) is not authorized either. These rights affect both the content of the thesis and its abstracts and indexes.

Lucas Marín Isern

DYNAMIC ADAPTATION OF USER PROFILES IN RECOMMENDER SYSTEMS

Ph.D. Thesis

Supervised by
Dr. Antonio Moreno and Dr. David Isern

Department of
Computer Science and Mathematics



UNIVERSITAT ROVIRA I VIRGILI

May, 2013

Acknowledgements

This work has been supported by Universitat Rovira i Virgili and the Spanish Ministry of Science and Innovation (DAMASK Project, Data mining algorithms with semantic knowledge, TIN2009-11005) and the Spanish Government (Plan E, Spanish Economy and Employment Stimulation Plan).

The author has been supported by a pre-doctoral research grant from the Universitat Rovira i Virgili.

I would like to thank the directors of this thesis, Dr. Antonio Moreno and Dr. David Isern for their guidance and support during the elaboration of this work.

Moreover, I also thank Dra. Aïda Valls and Dr. Josep M. Merigó for their collaboration, as well as the current and former ITAKA group members (Sergi, Carlos, Luis, Montse and David) for creating an excellent work environment during all this years.

Last but not least, I also want to thank my parents and my girlfriend for their trust and for encouraging me in the stressful moments.

Abstract

In a period of time in which the content available through the Internet increases exponentially and is more easily accessible every day, techniques for aiding the selection and extraction of important and personalised information are of vital importance. Recommender Systems (RS) appear as a tool to help the user in a decision making process by evaluating a set of objects or alternatives and aiding the user at choosing which one/s of them suits better his/her interests or preferences. Those preferences need to be accurate enough to produce adequate recommendations and should be updated if the user changes his/her likes or if they are incorrect or incomplete. In this work an adequate model for managing user preferences in a multi-attribute (numerical and categorical) environment is presented to aid at providing recommendations in those kinds of contexts. The evaluation process of the recommender system designed is supported by a new aggregation operator (Unbalanced LOWA) that enables the combination of the information that defines an alternative into a single value, which then is used to rank the whole set of alternatives. After the recommendation has been made, learning processes have been designed to evaluate the user interaction with the system to find out, in a dynamic and unsupervised way, if the user profile in which the recommendation process relies on needs to be updated with new preferences. The work detailed in this document also includes extensive evaluation and testing of all the elements that take part in the recommendation and learning processes.

Contents

Chapter 1	Introduction.....	1
1.1	Framework of this Ph.D. thesis.....	4
1.2	Objectives	4
1.3	Document structure.....	4
Chapter 2	User preferences.....	5
2.1	Preference Management.....	6
2.1.1	Profiles structure	6
2.1.2	Generating initial profiles	9
2.2	Preferences over numerical attributes.....	10
2.3	Preferences over categorical attributes	12
2.4	Profile composition.....	13
2.5	Alternative evaluation using the information in the profile ...	13
2.6	Conclusions.....	14
Chapter 3	Two new aggregation operators: ULOWA and IULOWA	17
3.1	Aggregation operators.....	18
3.2	Ordered Weighted Averaging (OWA) aggregation operators	19
3.2.1	Linguistic OWA operator.....	21
3.2.2	Induced OWA operator.....	21
3.3	Unbalanced LOWA operator	23
3.3.1	Preliminaries	24
3.3.2	Definition of the ULOWA operator	27
3.3.3	Properties of the ULOWA operator	28
3.3.4	Examples.....	30
3.3.5	Comparison with other linguistic operators	33
3.3.6	ULOWA Conclusions	38
3.4	Induced ULOWA operator	38
3.4.1	Definition of the IULOWA operator.....	39
3.4.2	Properties of the IULOWA operator.....	40
3.4.3	Families of IULOWA operators.....	41
3.4.4	Order Inducing Variables.....	42
3.4.5	IULOWA multi-person multi-criteria case study.....	49
3.4.6	IULOWA Conclusions.....	55
3.5	Conclusions.....	56
Chapter 4	Preference Learning.....	57
4.1	State of the art.....	58
4.1.1	Number of criteria.....	59

4.1.2	Use of domain-dependent information.....	60
4.1.3	Dynamic preferences.....	61
4.1.4	Management of user historical data	61
4.1.5	Final discussion.....	62
4.2	Preference learning over numeric preferences.....	64
4.2.1	On-line adaptation process	64
4.2.2	Off-line adaptation process	67
4.2.3	Evaluation	69
4.2.4	Preference function learning	79
4.3	Preference learning over categorical attributes	82
4.3.1	On-line adaptation process	83
4.3.2	Off-line adaptation process	85
4.3.3	Adaptation mechanism.....	87
4.3.4	Evaluation	88
4.3.5	Preference learning on multi-valued attributes	97
4.4	Numeric and linguistic simultaneous learning.....	100
4.5	Conclusions.....	100
Chapter 5	Case study: Restaurant recommendation	103
5.1	Barcelona restaurants data	103
5.2	Recommendation and adaptation	105
5.3	Results evaluation.....	106
5.4	Conclusions.....	112
Chapter 6	Conclusions.....	113
6.1	Contributions	113
6.2	Publications.....	115
6.3	Future work.....	118
References	121

List of figures

Figure 1. Modules of the recommender framework	3
Figure 2. Screen capture of the initial user information form in SigTur	10
Figure 3. Basic numeric preference function	11
Figure 4. Numeric preference function described with 5 parameters ($vpref$, ml , mr , Δl and Δr)	12
Figure 5. Examples of (a) balanced, and (b) unbalanced linguistic preference sets.....	12
Figure 6. User profile example	13
Figure 7. Evaluation of an alternative.....	14
Figure 8. Combinations used to analyze the similarity function.....	25
Figure 9. Graphical example of ULOWA when aggregating two labels (VL and P) with a mean average	27
Figure 10. Fuzzy term sets with 7 and 9 labels: (a) and (c) balanced; (b) and (d) unbalanced.....	31
Figure 11. Examples of ULOWA when aggregating VL and AH with an average weight.....	32
Figure 12. Unbalanced term set with 5 linguistic labels (b) obtained from a linguistic hierarchy of 3 levels (a).....	36
Figure 13. A set of nine linguistic labels (from (Xu 2009))	37
Figure 14. Examples of ULOWA aggregation of two labels (VL and P) when changing the weights.....	43
Figure 15. Two fuzzy sets with the same specificity and different fuzziness	46
Figure 16. Linguistic variable with 7 terms (test 1).....	46
Figure 17. Linguistic variable with 5 terms (test 2).....	48
Figure 18. Diagram of the multi-person multi-criteria aggregation process	50
Figure 19. Evaluation scale for the criteria (D: “Dangerous”, R: “Risky”, PO: “Poor”, A: “Acceptable”, G: “Good”, E: “Excellent” and PF: “Perfect”)	52
Figure 20. Attraction and repulsion forces, namely F_s and F_o respectively	65
Figure 21. On-line adaptation process	66
Figure 22. Off-line adaptation process.....	68
Figure 23. Evaluation of the adaptation algorithms	70
Figure 24. Influence of the parameter α	73
Figure 25. Influence of the parameter β	73
Figure 26. Distance between the ideal and the adapted profiles using the adaptation processes	74
Figure 27. Position of the user’s favourite option in the set of ordered alternatives.....	75
Figure 28. Evolution of the distances for each criterion.....	76
Figure 29. Evolution of the <i>nearest airport distance</i> (NAD) preference values	78
Figure 30. Evolution of the <i>average hotel price</i> (AHP) preference values	79
Figure 31. Preference function learning algorithm	80
Figure 32. Preference function learning steps.....	81
Figure 33. Preference learning process	82
Figure 34. Pseudo-code of the on-line process	84
Figure 35. Pseudo-code of the off-line adaptation process	86

Figure 36. Pseudo-code of the simulator	88
Figure 37. News attributes and possible values	90
Figure 38. Fuzzy label set with 5 balanced terms	91
Figure 39. Distances between the current and the ideal profiles	92
Figure 40. A quantitative study of the misclassified labels in the comparison of the current profile with the ideal profile	93
Figure 41. An empirical study of the influence of the characteristics extraction threshold.....	94
Figure 42. An empirical study of the influence of the maximum number of profile changes per iteration	95
Figure 43. An empirical study of the influence of the level of evidence required to make a change	95
Figure 44. An empirical study of the influence of the minimum number of stored selections.....	96
Figure 45. An empirical study of the influence of the minimum number of over-ranked alternatives	97
Figure 46. Example of ranked alternatives	99
Figure 47. Independent execution of the adaptation processes.....	100
Figure 48. Restaurants data details	104
Figure 49. "Distance to city center" values distribution	104
Figure 50. "Average price" values distribution.....	105
Figure 51. Linguistic preference label set of 7 values	105
Figure 52. Three initial profiles and ideal profile used in the evaluation.	106
Figure 53. Average distance between the current and the ideal profile	107
Figure 54. Learning evolution of the preference over three values of the attribute "Types of food"	108
Figure 55. Learning evolution of preferences on the values of the attribute "Atmosphere"	108
Figure 56. Learning evolution of the preferences on the values of the attribute "Special characteristics"	109
Figure 57. Evolution of the <i>vpref</i> value of the attribute "Average price"	110
Figure 58. Evolution of the <i>vpref</i> value of the attribute "Distance to city center"	110
Figure 59. Evolution of "Distance to city center" numeric preference function.....	111
Figure 60. Position of the selected alternative in each iteration (Test 1)	112

List of tables

Table 1. Comparison of similarity functions between fuzzy sets	26
Table 2. Comparison of LOWA and ULOWA operators	34
Table 3. Uncertainty measures for the terms in Figure 16	46
Table 4. Weights obtained without specificity	48
Table 5. Weights obtained in Test 1	48
Table 6. Weights obtained in Test 2	49
Table 7. Environmental criteria	50
Table 8. Definition and values of specificity and fuzziness of the linguistic terms	52
Table 9. Evaluation of expert A	53
Table 10. Evaluation of expert B	53
Table 11. Evaluation of expert C	53
Table 12. Collective data matrix, including the overall suitability value	54
Table 13. Set of criteria of the analysed alternatives	71
Table 14. Preference changes at each iteration (average of 10 tests)....	77
Table 15. Preference changes every 5 iterations (average of 10 tests) .	77
Table 16. Preference changes every 10 iterations (average of 10 tests)	77
Table 17. An example set of rated and sorted alternatives.....	82
Table 18. Linguistic preference adaptation process parameters	83

Chapter 1

Introduction

Living in the so called *knowledge society* means that we are constantly in contact with ways that facilitate our access to unlimited sources of information and knowledge that help us in our daily activities. Moreover, as it is easy to reach that immense source of data, it is also easy to create and publish new content and make it available for others. Nowadays, with the explosion of the social networks and tools that take advantage of their infrastructure and reach, the task of creating and making new content available such as events or opinions about all sorts of things has also been eased for all sectors of population, including the ones that, until now, were not very familiarized with the information technologies. This fact allowed an exponential increase of the content available but, inevitably, worsened an already existing problem: the created content is intrinsically heterogeneous and unstructured; there is no control over it in format neither in content, so it is frequent to find redundant and/or incomplete information.

In this scenario, we are constantly confronted with situations in which we rely on the information technologies to find solutions to decision problems in which we have to evaluate a considerable set of possible options. Those daily situations, such as deciding which radio station to listen to, which program to watch in TV, or the location to go on holidays, are solved taking into account our own preferences to rate all the alternatives we have. Imagine the case of choosing a place to spend our holidays. Almost every user who wants to plan a travel nowadays uses Internet to find information about possible destinations, searching through thousands of Web pages, trying to find useful content to help him decide at which place he/she should go. After a search, the user identifies locations that he/she is interested in, and evaluates them taking into account some properties that identify each destination, named *criteria*, according to his/her interests or *preferences*. Due to the explosive growth and variety of the information available noted previously, this task frequently becomes an overwhelming and time consuming one if we do it by ourselves.

The *recommender systems* (Resnick, Varian 1997) appeared with the objective to help the user in that process of evaluating different options or items, and aiding him/her to choose the one that best fits his/her interests. This recommendation task can be done in different ways, which is used to distinguish between different types of systems (Burke 2007):

- *Content-based*: Recommend items that are similar to the ones that the user liked in the past.
- *Collaborative filtering*: Recommend to the active user items that were liked in the past by similar users.

- *Demographic*: Recommend items based on the demographic profile of the user.
- *Knowledge-based*: Recommend items based on specific domain knowledge about how certain item features meet users' needs and preferences and, ultimately, how the item is useful for the user.
- *Community based*: Recommend items based on the preferences of the user friends.
- *Hybrid*: Combinations of the other mentioned techniques.

In recent years, as noted in (Ricci et al. 2011), the interest in recommender systems has dramatically increased, as the following facts indicate:

- They play an important role in such highly rated Internet sites as *Amazon, YouTube, Netflix, Yahoo* or *Tripadvisor*.
- There are dedicated conferences and workshops related to the field, being the most representative the *ACM Recommender Systems (RecSys)*, established in 2007.
- There are graduate and undergraduate courses dedicated entirely to recommender systems at institutions of higher education around the world.
- There have been several special issues in high impact academic journals covering research and developments in the field, such as *AI Communications, IEEE Intelligent Systems* or *ACM Transactions on Information Systems*.

The accuracy of the recommendations mainly depends on three elements: the *knowledge* the system has about the user interests or preferences, how it *exploits* that information to drive the recommendation process, and the capacity of the system to *learn* or update this knowledge.

When the system performs the ranking of the items that can be recommended to the user, the information stored about his/her interests is used with the objective to leave at the beginning of the ranked list the elements that fit better the user requirements. That stored information is called *user profile*, and is a data structure which contains relevant information about the user likes regarding the items in the recommendation problem. User profiles, as is seen deeply in Chapter 2, can be structured in various ways. For example, in a collaborative filtering system, a user profile consists in a simple list containing the ratings provided by the user for some items, while in a demographic recommender system, socio-demographic attributes such as age, gender or location are used to build the profile. The system presented in this Thesis relies on a user profile which contains preferences about the numerical and categorical criteria (*attributes*) that define the items or *alternatives*.

As said above, recommender systems use the information stored in the profile to decide which alternatives are going to be recommended to the user. More advanced systems can sort that list of alternatives by descending order of satisfaction, as the one developed in this Thesis, being the first option the one that the system finds to be the most suitable to the user. This ranking is done by evaluating each option and giving it a level of satisfaction with respect to the user. In the approach explained in this document, in which the alternatives are defined by multiple attributes,

preferences are used to evaluate each single attribute value, using a linguistic scale, in terms such as “Low”, “Medium” or “High”. Then, all of those preference terms are combined into a single one in a process called *aggregation*, as explained in Chapter 3.

Traditional recommender systems have two ways of functioning: *supervised* and *unsupervised*. In *supervised* systems the user has access to his/her preferences in order to change them. This method allows knowing the exact preferences of the user, but it is a time-consuming task that usually users are reluctant to make. For this reason in this Thesis the main effort has been put at adapting the personal profile of the users in an *unsupervised* way, in which the user interests are unknown and they are initialised and updated automatically. Moreover, it is assumed that the system is used regularly, so that it can notice, from the user interaction over time, if and how the preferences of the user evolve. Learning processes developed for this work, explained in Chapter 4, are divided in two kinds: the ones that are executed at each interaction with the user (called *on-line* processes), and the ones that are executed after a certain number of interactions and use historical data from the user interaction with the system (called *off-line* processes).

To sum up, the final purpose of this thesis is the design of a self-customising framework that permits the acquisition of knowledge about the tasks the user performs in order to filter large amounts of possibilities, rate them and present them in a sorted way to the user. Moreover, the system is able to evolve the user interests from his/her interaction with it. The framework is designed in two separate modules: rating and ranking a set of alternatives and the adaptation of the user profile (see Figure 1).

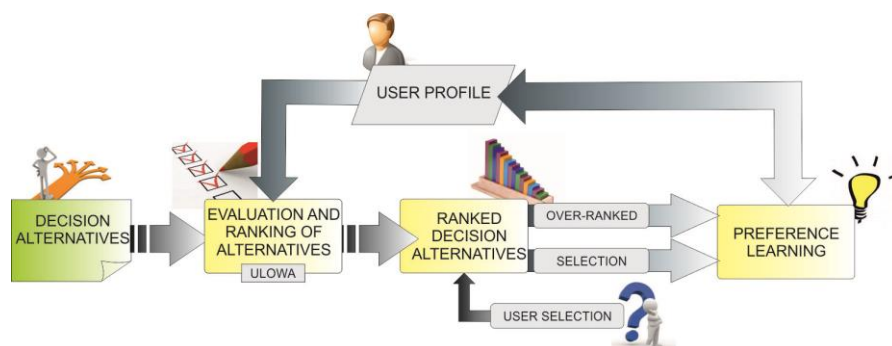


Figure 1. Modules of the recommender framework

The functioning of the whole system designed in this work is the following:

- 1) The recommender system receives a set of alternatives composed by several criteria.
- 2) The user profile stores the preferences of the user over all those criteria.
- 3) Afterwards, the task of the recommender is to rate all the alternatives taking into account the user profile in order to prioritize them according to his/her particular circumstances.
- 4) Then, the user selects the most appropriate alternative from this sorted list.
- 5) Finally, the adapting algorithm collects the information provided by this selection, the set of unselected alternatives,

and all past selections, to infer which changes can be made to the current profile.

1.1 Framework of this Ph.D. thesis

This work is part of the DAMASK (Data mining algorithms with semantic knowledge, TIN2009-11005) research project funded by the Spanish Ministry of Science and Innovation and Universitat Rovira i Virgili. The work has been supported by the Spanish Government (Plan E, Spanish Economy and Employment Stimulation Plan).

The author of this Thesis has been supported by a pre-doctoral grant of Universitat Rovira i Virgili.

1.2 Objectives

The main objectives of this thesis can be summarized as follows:

1. To build a generic knowledge-based framework that permits to acquire different kinds of data (mainly numerical and linguistic), which can be used to model a *user profile*.
2. To create a decision-ranking procedure to evaluate and rank sets of alternatives. This process should deal with different kinds of variables.
3. To create a flexible and dynamic mechanism to obtain adequate recommendations by adapting the preference functions over time in an unsupervised way by observing the user interaction with the system.

1.3 Document structure

This document is divided into the following chapters:

- Chapter 2 presents the concept of user preference or interest and how these numerical and categorical preferences can be represented and managed.
- Chapter 3 details the utility and necessity of the aggregation operators and introduces the operators designed during the elaboration of this thesis.
- Chapter 4 discusses the notion of preference learning and presents a method for learning user preferences over numeric and categorical attributes by evaluating the user interaction with a recommender system.
- Chapter 5 evaluates the designed learning techniques by using a real dataset of restaurants of the city of Barcelona defined by numeric and categorical attributes.
- Chapter 6 discusses the conclusions and identifies some future lines of research.

Chapter 2

User preferences

As seen in the introduction, the basis of a recommender system is the evaluation and ranking of a set of alternatives, which are represented through a set of attributes or criteria, taking into account the user's interests. In order to do this, it is necessary to know the user preferences about the values of the different attributes (numeric and categorical) that are used to represent the domain objects (alternatives). Then, a way to evaluate the alternatives in base to those preferences needs to be designed in order to see which ones fit better the user requirements. This chapter is focused on the first part: the definition and management of user preferences over numeric and categorical attributes.

Öztürk *et al.* (Öztürk et al. 2006) defines a *decision problem* as the case of somebody who tries to compare objects taking into account different points of views or criteria. Comparing two objects can be seen as looking for one of the following situations:

- Object a is “before” object b , where “before” implies some kind of order between a and b , with such an order referring either to a direct *preference* (a is preferred to b) or being induced from a measurement and its associated scale (a occurs before b , a is longer, bigger, more reliable, than b).
- Object a is “near” object b , where “near” can be considered either as indifference (object a or object b will do equally well for some purpose), or as a similarity, or again could be induced by a measurement (a occurs simultaneously with b , they have the same length, weight, reliability).

From a decision aiding point of view we traditionally focus on the first situation. Ordering relations is the natural basis for solving ranking or choice problems. The second situation is traditionally associated with problems where the aim is to be able to put together objects sharing a common feature in order to form “homogeneous” classes or categories (a classification problem).

As stated in (Öztürk et al. 2006; Fürnkranz, Hüllermeier 2003), given a set of alternatives A , establishing how each element of A compares to each other element of the same set from a *preference* point of view enables to obtain an order which might be used to make either a choice on the set A (identify the best) or to rank the set A . It is necessary to consider whether it is possible to establish such an ordering relation and of what type for all pairs of elements of A . It is also necessary to establish what the meaning of the lack of preference is.

As pointed out before in this document, we use a linguistic scale to express the level of preference of each of the criteria values of the alternatives that the recommender system evaluates (such as *Very low*, *Low*, *Medium*, *High* and *Very high*).

Those preferences, that define the level of interest of a user in a concrete object, are usually stored in a data structure known as *profile*. Through this document, it has been considered as *profile learning* the action to extract enough information from a user to build a profile of him/her, accurate enough to start making correct recommendations, including the actions aimed to update the current user profile as that user preferences change over time.

One of the goals of the Thesis is to create a framework able to deal with different types of attributes. To do that, the first step was to create a framework to work with *numerical* attributes (*e.g.*, temperature). When this goal was achieved, a more complete version of the system permitted the management of categorical attributes (*e.g.*, languages). Finally, the last step was the creation of a powerful and generic framework able to combine both types of criteria.

Section 2.1 includes a summary of the different ways to represent the information about the user or about his/her preferences in the user profile, and how the associated data structures can be initialised. Further in Chapter 4 it will be seen how those preferences can be dynamically modified or adapted through observing the user interaction with the system. Sections 2.2 and 2.3 show how it has been decided to represent the information regarding preferences in the system developed in this Thesis. Specifically, section 2.2 describes the representation of the preferences about numerical attributes, and section 2.3 explains how preferences over categorical attributes are represented. Section 2.4 shows how preference information about the two types of attributes is integrated in the user profile. Finally, section 2.5 includes an explanation of how the alternatives are evaluated in the system.

2.1 Preference Management

This section presents a state-of-the-art on preference management which includes a study on techniques to manage user information by means of *profiles*: the definition of its representation (subsection 2.1.1) and its initial generation (subsection 2.1.2).

2.1.1 Profiles structure

As briefly indicated in the previous section, the first step to design a preference-based system is to define data structures to store information about the users' interests; the whole collection of interests constitutes the user profile.

Stored information

User profiles normally contain two types of information: demographic characteristics and domain-dependent preferences (Tso, Lars 2006).

Demographic characteristics, such as civil status, age or studies, identify the user in a social group or domain. This kind of information is usually not enough to drive the recommendation processes. However, it can have some utility in certain situations:

- a) Provide an insight about the “type” of user we are dealing with, fact that can allow to make approximate initial recommendations that can be improved and refined dynamically through the interaction with the user (Moreno et al. 2013).
- b) Classify the user into a predefined group of user class prototypes (stereotypes), to which a set of experts have assigned their ideal preferences (Rich 1979).
- c) Compare user demographical information to start giving generic recommendations based in users with similar social characteristics, as done in (Basiri et al. 2010). This information is usually employed in collaborative systems.

On the other hand, most of the profiles contain information of the context, called *domain-dependent preferences*. As (Eyharabide, Amandi 2012) explain, some works consider a limited version of the context, in which it is defined informally, generally known in advance, and determined in a fixed way. The goal is to isolate the most promising attributes.

This Thesis, however, focuses on using *domain-dependent preferences*, in which we deal with characteristics of the objects to recommend and apply content-based recommendation mechanisms according to the individual preferences of a single user over those characteristics.

Object-based classification

There are several ways in which preferences may be represented in the user profile. A first classification can be found in (Montaner 2003), and represents a way of categorising preference management models more focused on the objects that intervene in the whole procedure than on the preferences themselves: history-based model, vector space model, weighted n-grams, semantic networks, classifier-based models, and user-item ratings matrix.

A profile following the *history-based* model stores a list of activities resulting from the interaction between the user and the system (Mianowska, Nguyen 2013; Rastegari, Shamsuddin 2010). A good example of this type of systems is *Amazon*, that stores a list of purchases that complement the information provided by the user, which finally personalises the online store for each customer (Jannach 2006). The main disadvantage in that model is that a lot of space is required to store historical data of each user and, therefore, the processing time to evaluate it all is very high. Moreover, systems that implement that model of representation are not very generic so history-based information is difficult or almost impossible to translate and use through different domains.

In the *vector space model*, preferences are extracted from items (usually documents) which are represented with a vector of features (terms) with an associated value. This value can be a Boolean (which indicates the presence of the feature) or a real number (which indicates the frequency, relevance or probability of the feature). As an example, WebMate (Chen, Sycara 1998) utilises a multiple vector representation in which the basic idea is to represent each document as a vector in a vector space so that documents with similar content have similar vectors. Each dimension of the vector space represents a word and its weight. This model provides a concise representation of the domain objects, facilitating their comparison and/or their classification in groups according to some similarities. Normally, preferences can be represented in the same way, with the user preferred value for each feature/term. In this way, it is easy to compare each object with the user preferences and apply content-based recommendation mechanisms.

In *weighted n-grams*, items are represented with a net of words with weights in the nodes and edges. This technique is applied in PSUN (Sorensen, McElligot 1995). Relying on the idea that related words tend to occur one after another a significantly high number of times, fixed length consecutive series of n characters are extracted and organised with weighted links representing the co-occurrence of different words. Therefore, the structure achieves a contextual representation of the words.

Semantic networks are able to represent and store semantic relations and meaning among concepts. In ifWeb (Minio, Tasso 1996) semantic networks are used to describe typical patterns of topics of interest to the users. Other examples presented by (Eyharabide, Amandi 2012) and (Blanco-Fernández et al. 2011) show methods that permit to dynamically link preferences to concepts of the ontology.

Systems which learn from the user using some sort of *classifier* as learning technique retain the structure of the classifier itself as the user profile. Examples of that kind of profiles are neural networks (Boone 1998), decision trees (Krulwich, Burkey 1996), induced rules (Basu et al. 1998) and Bayesian networks (Jensen 1996).

Some systems maintain a matrix which stores *user ratings* on items (Marlin 2003). Each position of this matrix (u,i) contains the rating done to item i by user u . This ratings can be expressed using a linguistic scale (terms like “Poor”, “Normal” or “Good”), a numeric scale (like giving an evaluation from 0 to 10), or a symbolic scale (using stars). This kind of information is usually managed in recommendation procedures based on collaborative filtering, where users that give similar ratings to the same items are grouped.

It can be noted that, in the last two models, preferences are stored implicitly (e.g. weights inside the neural network or punctuation given to an item) rather than expressed explicitly.

Preference-based classification

Preference modelling techniques can also be classified according to the kind of preferences that are stored, rather than in the representation of the domain objects:

- a) User profiles that contain a vector of values in which the user is interested (Phelan et al. 2011a).
- b) User profiles that contain qualitative preferences represented with fuzzy terms (Serrano-Guerrero et al. 2011; Morales-del-Castillo et al. 2009; Morales-del-Castillo et al. 2010).
- c) User profiles containing preferences on numerical criteria (Joachims, Radlinski 2007).
- d) User profiles linked with ontology-based semantic information (Arias et al. 2011; Moreno et al. 2013; Noppens et al. 2006).

Note that most of these options (both in the first and in the second classification) focus on a single kind of preference values, which is either numeric (e.g. the user's assessment of a particular object or the user's level of interest in objects related to a certain class of the domain ontology) or linguistic (e.g. the user may be requested to fill a questionnaire describing his/her interests using some predefined linguistic labels, so that he/she is not forced to give a precise numerical value). The joint consideration of quantitative and qualitative preferences on numerical and multi-valued categorical attributes proposed in this Thesis is not directly addressed in this classification and provides an added value to the previous works in the field.

2.1.2 Generating initial profiles

After defining how the user preferences are stored it is necessary to think about how to get the initial data for the profile in order to start using it in the recommender system. The lack of initial information about the user profile (or about the user preferences over the recommending items) is known in the literature as *cold-start problem*, and there are several ways to manage it (Schein et al. 2002).

Some systems allow starting to work with an *empty profile* structure which is filled through an automatic recognition method when the user begins to interact with the system (Lee, Rho 2012).

A system can also require from the users a *manual registration* of their interests. This implies more work for the user and it is not accurate, since some interests may still be unknown at the moment the user starts to use the system. For example, in (Moreno et al. 2013), the user fills an initial form (see Figure 2) with his/her interests about the classes in the first level of a touristic ontology. Then, the ontology structure is used to transmit that information to the rest of the concepts of that ontology in an adequate way.

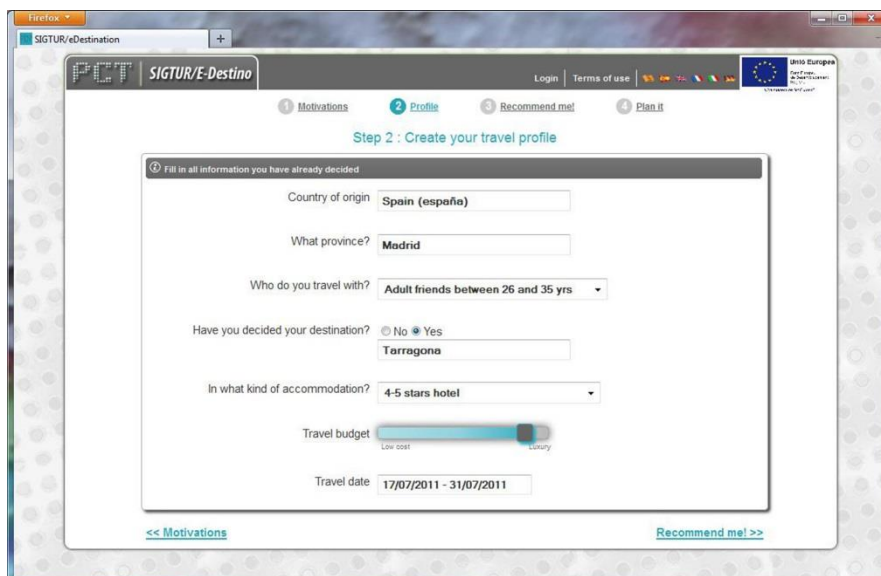


Figure 2. Screen capture of the initial user information form in SigTur

The initial user profile can also be modelled by *stereotyping*. Users can be classified in stereotypical descriptions which represent the features of classes of users. Demographic data which the user may indicate in a registration form is used to classify him/her in a group that has a predefined set of preferences configured by a consensus of domain experts. Stereotyping methods can associate each new user to a single group or, as done in more complex systems, a percentage of similarity to each of the predefined groups can be computed. This method is implemented in the LifeStyle Finder (Krulwich 1997), which uses a commercially available database of demographic data which encompasses the interests of people nationwide. The main problem of this technique is that users usually provide incomplete or false data so the stereotyping is inaccurate in those cases.

Finally, another method for modelling the initial profile is using a *training set*. It contains a set of user interaction examples which are shown to the user, such as a list of products to rate. User interaction is then processed with a learning technique in order to generate the profile. For example, in (Sun et al. 2013) the user is initially faced with an interview process guided by a decision tree in which he/she is directed to child nodes depending on his/her responses. Finally, an algorithm uses the answers as input to predict item ratings.

2.2 Preferences over numerical attributes

Numerical attributes are properties which are represented using a numerical value. An example of that kind of attributes could be the property “Population density” when describing a “Tourist destination”, that could take the value “200 people per km²”.

For that kind of attributes, the profile contains a value v_{pref} that represents the preferred value of the user in the domain of the attribute. In order to evaluate the degree of preference of any value of the attribute, in our initial works (Marin et al. 2011b) it was assumed that each user has a

preference function for each attribute, which has a triangular shape (see Figure 3) and is defined as:

$$p_a(x) = \begin{cases} 0 & \text{if } (x < (v_{pref} - \Delta)) \\ 1 - \frac{|x - v_{pref}|}{\Delta} & \text{if } ((v_{pref} - \Delta) \leq x \leq (v_{pref} + \Delta)) \\ 0 & \text{if } (x > (v_{pref} + \Delta)) \end{cases} \quad (2.1)$$

where $p_a(x)$ is the preference of the value x of the attribute a and Δ is the width of the function, which it was considered to be 10% of the attribute domain. Note that the only point with a preference 1 is precisely v_{pref} , and that any point below $v_{pref} - \Delta$ or above $v_{pref} + \Delta$ has preference 0.

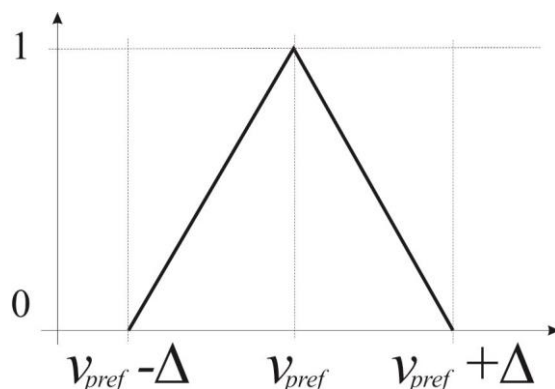


Figure 3. Basic numeric preference function

This approach to numerical preference function, however, lacks expressivity since the slopes of the function cannot be expressed (is implicitly assumed to be 1) and the widths in both sides of v_{pref} are symmetrical and fixed (Δ). So a new approach to represent and manage preference functions was designed. It has 5 parameters (left and right slope, left and right width, and value of maximum preference) instead of considering only the preferred value. Its definition is as follows:

$$p_a(x) = \begin{cases} 1 - \left(\frac{|x - v_{pref}|}{\Delta_l} \right)^{m_l} & \text{if } (x < v_{pref}) \\ 1 & \text{if } (x = v_{pref}) \\ 1 - \left(\frac{|x - v_{pref}|}{\Delta_r} \right)^{m_r} & \text{if } (x > v_{pref}) \end{cases} \quad (2.2)$$

In this expression, $p_a(x)$ is the preference of the value x of the attribute a , m_l and m_r are the slope values (for the left and right sides of the function, respectively) and Δ_l and Δ_r define the width of the function (also for the left and right sides, respectively).

Figure 4 shows an example of a preference function, where the left slope is a value under 1 (concave slope), the right slope is a value over 1 (convex slope), and the left width is greater than the right one.

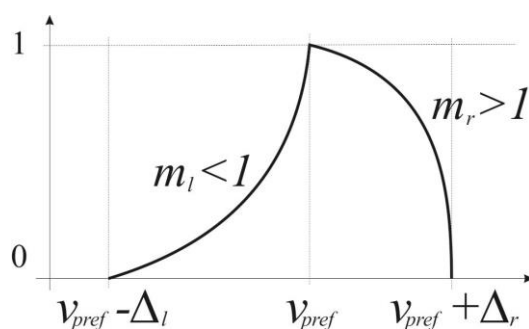


Figure 4. Numeric preference function described with 5 parameters (v_{pref} , m_l , m_r , Δ_l and Δ_r)

2.3 Preferences over categorical attributes

Categorical attributes are properties that are represented using one or a list of linguistic values or categories. One example of that kind of property could be the attribute “Native Language” when describing a “Tourist Destination” that could, for example, take the values “Spanish and English”.

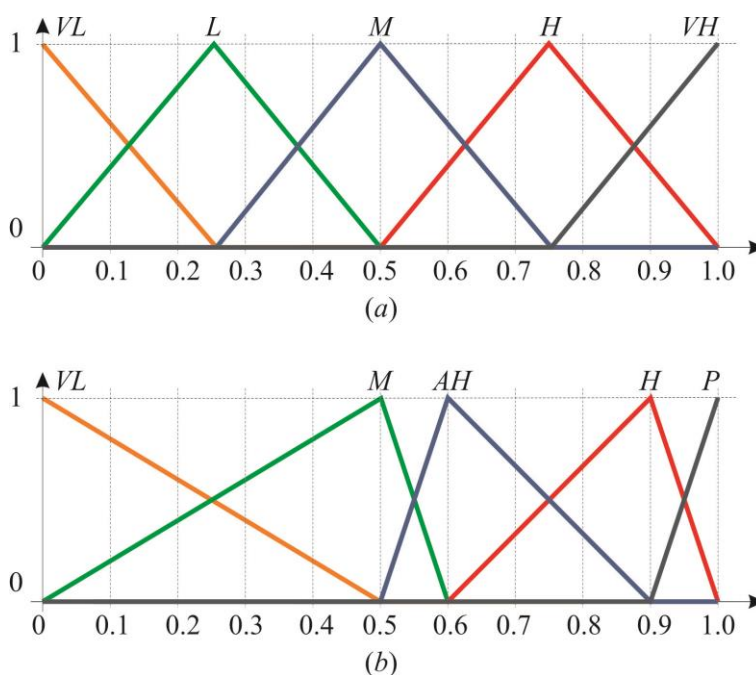


Figure 5. Examples of (a) balanced, and (b) unbalanced linguistic preference sets

For this kind of attributes, a linguistic level of preference has to be indicated for each possible value or values that the attribute could take. In a recent work (Marin et al. 2013) it has been proposed to represent the level of interest over each value in the domain of the categorical attributes by using a linguistic scale in which the semantics of preference labels is defined using fuzzy sets (see Figure 5a with an example with the labels “Very Low”, “Low”, “Medium”, “High” and “Very High”). This example shows a set of linguistic labels that are symmetrical and uniformly distributed. However, in some situations it can be more appropriate to represent the preferences using fuzzy sets that are not symmetrical or are not distributed uniformly through the domain, as shown in Figure 5b with

the label set (“Very Low”, “Medium”, “Almost High”, “High” and “Perfect”).

2.4 Profile composition

Figure 6 represents an example of a user profile which combines numerical (left side) and categorical (right side) attributes. The numerical preferences define the values of maximum preference of two attributes v_{pref_1} and v_{pref_2} over the numerical attributes a_1 and a_2 , respectively, as well as the values of right and left slopes (m_l and m_r) and widths (Δ_l and Δ_r) of the numerical preference function of each attribute; those values belong to the particular domain of each variable. On the other hand, the right part contains the qualitative preferences over the possible values of the categorical attributes a_3 and a_4 , represented with the linguistic labels depicted in Figure 5a.

User Profile					
Preferences on numerical attributes				Preferences on categorical attributes	
a_1		a_2		a_3	a_4
v_{pref_1}	23	v_{pref_2}	23	v_{3a}	VH
Δ_{l1}	10%	Δ_{l2}	10%	v_{3b}	L
Δ_{r1}	15%	Δ_{r2}	30%	v_{3c}	H
m_{l1}	0.8	m_{l2}	1	v_{3d}	M
m_{r1}	2	m_{r2}	1.5	v_{4a}	VH
				v_{4b}	VL
				v_{4c}	M
				v_{4d}	L
				v_{4e}	H

Figure 6. User profile example

2.5 Alternative evaluation using the information in the profile

The main purpose of the user profiles is to use the information stored inside them to study how the possible alternatives the user faces fit his/her interests. So, when evaluating an alternative, the objective is to aggregate all of the values of preference assigned to each of the values of its attributes into a single value. Since two kinds of attributes are being considered, a conversion to the same domain is made.

In the approach studied in this Thesis, it has been chosen to translate the numerical preferences into linguistic ones. The translation is done by, first, calculating the value of preference of every numeric attribute using its numeric preference function. Then that value is mapped to the fuzzy linguistic label with a higher value in that point, rounding it up to the greater label in the cases where the values are exactly in the middle point between two labels. For the case of the values of the categorical attributes, they are translated directly into their associated linguistic preference values in the profile.

When all the attribute values have been assigned a value of preference using the same fuzzy linguistic scale, all the terms are combined or *aggregated* using aggregation operators, which are introduced in the next chapter. The final result of that evaluation is the value of preference assigned to the whole alternative, which is used to rank the whole set of alternatives.

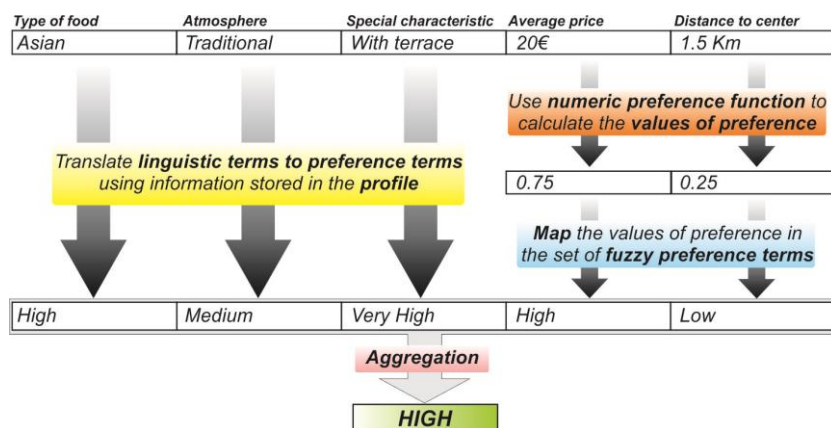


Figure 7. Evaluation of an alternative

The whole process of evaluation of an alternative is depicted in Figure 7, where an example of an Asian restaurant is evaluated according to a certain user profile. It includes three categorical attributes (“Type of food”, “Atmosphere” and “Special characteristic”) and two numerical ones (“Average price” and “Distance to centre”).

The preference terms (taken from the set shown in Figure 5a) for the first set of attributes are directly obtained from the user profile and they do not require any further interpretation. However, in the case of the values of the numerical attributes there is an intermediate step: first the numerical values are translated to a value of numerical preference using the attribute preference function, and then that value is translated to a linguistic term of preference. At this moment, as shown at the bottom, we have a list of linguistic labels that represent the user’s qualitative preferences on all the values of the alternative that is being evaluated. The final overall evaluation of the alternative is obtained with the application of an aggregation operator, resulting in the “High” label.

2.6 Conclusions

This chapter described how user preferences about numeric and categorical attributes are represented in the profile structure considered in the recommendation and learning processes designed in this Thesis.

The definition of a numerical preference function with five parameters for the expression of preferences over numeric attributes allows a high level of precision for representing the user satisfaction with the values of this kind of attributes.

In the case of categorical attributes, linguistic labels of preference such as “Low”, “Medium” or “High” are used to express the levels of interest about each possible value of the attribute. Those linguistic preferences are

defined as a set of linguistic labels, each one representing a value of preference. A linguistic preference scale permits to deal with the inherent uncertainty in the evaluation of the preferences in categorical variables.

Then, an example of the resulting profile structure containing attributes of both types has been shown. Finally, the process of evaluating an alternative has been explained, introducing the notion of *aggregation operator*, which is the focus of the next chapter.

Chapter 3

Two new aggregation operators: ULOWA and IULOWA

Every system which is able to aid in a decision problem, as introduced in the previous chapter, consists basically in the evaluation of the possible options considering the user interests or preferences. Those alternatives are considered over all this work as composed by multiple attributes, that is, many attribute values (numeric and categorical) describe a single alternative. As has been also explained in the previous chapter, every attribute value is translated to a level of preference according to the user profile. When the alternative is fully translated to linguistic preference terms, it is necessary to combine or *aggregate* all of them seeking to give a single value which gives an overall evaluation to the alternative with respect to the user, and can be used to sort the whole set of options. The tool used in that process is known as *aggregation operator*.

The first section in this chapter introduces the concept of aggregation operators, explaining the types of aggregation operators that can be found in the literature. In the second section, the family of the Ordered Weighted Aggregation operators is introduced, making a special emphasis on the ones in which the contributions in this area of this Thesis are based in (LOWA and IOWA operators). Sections three and four present the two contributions in the field of aggregation operators done during the development of this Thesis, respectively: the Unbalanced Linguistic Ordered Weighted Averaging (ULOWA) and the Induced Unbalanced Linguistic Ordered Weighted Averaging (IULOWA) operator. The last section also includes a case study of the new IULOWA operator applied to a multi-criteria multi-person environmental decision problem.

3.1 Aggregation operators

As pointed out by (Yager 1988), the problem of aggregating criteria functions to form overall decision functions is of considerable importance in many disciplines. As a response to that problem, aggregation operators appeared as functions that combine a set of values into a single one. For the purpose of this thesis, aggregation operators are used to aggregate all the preferences over the attribute values which define an alternative, with the objective to obtain a single linguistic/qualitative valuation of the whole alternative.

These operators, which are formally described as follows, typically satisfy the unanimity (idempotency) and monotonicity properties.

Definition 3.1. Let $a = (x_1, \dots, x_m)$ be a set of values in a domain D , and let $C: D^m \rightarrow D$ be a consensus function defined in a given domain D to aggregate the values x_1, \dots, x_m . The function should satisfy the following properties:

- a) Monotonicity: $C(x_1, \dots, x_m) \leq C(x'_1, \dots, x'_m)$ if $\forall i 1 \leq i \leq m, x_i \leq x'_i$, where \leq is an ordering relation in D .
- b) Commutativity: $C(x_1, \dots, x_m) = C(x'_1, \dots, x'_m)$ if (x'_1, \dots, x'_m) is any permutation of the elements of (x_1, \dots, x_m) .

Other properties such as continuity, boundary conditions, associativity or neutral element can also be satisfied by an aggregation operator.

There exist a large number of aggregation operators applicable to a broad range of data representation formalisms including ordinal and nominal scales (Xu, Da 2003). In general, aggregation operators can be classified according to the data type they handle (numerical, fuzzy, qualitative and heterogeneous) or according to their mathematical properties. The main families of aggregation operators are (Beliakov et al. 2007; Torra, Narukawa 2007; Yager 1988):

- Means (averaging functions), like the arithmetic mean, the weighted mean, the geometric mean, or the harmonic mean.
- Medians, which try to find a value that is more representative of a typical value than the mean. They essentially discard very high and very low values.
- Ordered weighted averaging functions (OWA), which are also averaging aggregation operators which associate weights not with a particular input, but rather with its value. According to the nature of the data, numerical or linguistic, OWA or LOWA operators can be defined, respectively.
- Choquet and Sugeno integrals, which are two classes of averaging functions defined with respect to a fuzzy measure. They are useful to model interactions between the criteria.
- Conjunctive and disjunctive functions, like the so-called triangular norms and conorms respectively. Minimum and maximum functions, product and probabilistic sum, Lukasiewicz norms, or drastic sum and product, are several examples of these aggregation functions that are used in fuzzy set theory and fuzzy logic.

- Mixed aggregation, used in situations where high input values are required to reinforce each other, whereas low values pull the output down. In this case, the aggregation function has to be disjunctive for high values, conjunctive for low values, and perhaps averaging if some values are high and some are low.

Due to their spread in the area of aggregation operators and their flexibility to accept modifications through which new operators can be defined, the Ordered Weighted Averaging (OWA) family of operators has been explored in this work.

3.2 Ordered Weighted Averaging (OWA) aggregation operators

The most important factor when determining the structure of the aggregation functions is the relationship between the criteria whose values we want to aggregate. At one extreme there is the situation in which we desire that all the criteria are satisfied. At the other extreme there is the case in which the satisfaction of any of the criteria is all we desire. These two extreme cases lead to the use of “and” and “or” operators (respectively) to combine the criteria functions.

The family of Ordered Weighted Averaging (OWA) operators was defined in (Yager 1988) with the objective to provide a range of aggregation operators which lie in between these two extremes, that could be called “orand” operators. An example of an operator of this new family could be the simple mean. It can be said that the OWA operator allows to easily adjust the degree of “anding” and “oring” implicit in the aggregation, and it is defined as follows:

Definition 3.2. A mapping F from $I^m \rightarrow I$ (where $I = [0,1]$) is called an OWA operator of dimension m if associated with F , is a weighting vector W , such that

- 1) $w_i \in [0,1], \forall i \in 1..m$
- 2) $\sum_i w_i = 1$

and where $F(a_1, a_2, \dots, a_m) = w_1 b_1 + w_2 b_2 + \dots + w_m b_m$, where b_i is the i -th largest element in the collection (a_1, a_2, \dots, a_m) .

We shall call a vector B of length m an ordered argument vector if each element $b_i \in [0,1]$ and $b_i \geq b_j$ if $j > i$. Given an OWA operator F with weight vector W and an argument tuple (a_1, a_2, \dots, a_m) we can associate with this tuple an ordered input vector B , such that B is the vector consisting of the arguments of F put in descending order. Using this notation then $F(a_1, \dots, a_m) = W \cdot B$, the inner product of W and B . It is important to emphasize the fact that the weights, the w_i 's, are associated with a particular ordered position rather than with a particular element. That is, w_i is the weight associated with the i -th largest element whichever component it is.

The following example illustrates the use of the OWA operators to aggregate a set of values considering a weight vector:

Assume F is an ordered weighting averaging operator of size $m = 4$ with weighting vector $W = (0.2, 0.3, 0.1, 0.4)$ and we want to aggregate the values in $A = (0.6, 1, 0.3, 0.5)$.

In this case the ordered argument vector is $B = (1, 0.6, 0.5, 0.3)$, hence $F(0.6, 1, 0.3, 0.5) = W \cdot B = (0.2)(1) + (0.3)(0.6) + (0.1)(0.5) + (0.4)(0.3) = 0.55$.

The weight vector specifies the decision-maker policy, so that we can emphasize different arguments based upon their ordered position. The weight vector can be generated according to a selected policy or *linguistic quantifier*.

The classic binary logic allows for the representation of two quantifiers, “there exists” and “for all”. As noted in (Yager 1988), in natural language we use many additional quantifiers such as “almost all”, “few”, “many”, “most”, etc. As indicated by Yager, the use of a linguistic quantifier in multi-criteria decision making provides a deeper and more unifying interpretation of the weighting function associated with an aggregation operator.

Yager indicated that the weighted vector W is a manifestation of the quantifier underlying the aggregation process. In particular, if a decision-maker wants Q of the objectives satisfied, then we obtain the weighting vector as $w_i = Q(i/m) - Q((i-1)/m), i = 1, \dots, m$, being the membership function of Q , as follows ($a, b, r \in [0,1]$):

$$Q(r) = \begin{cases} 0 & \text{if } r < a \\ \frac{r-a}{b-a} & \text{if } a \leq r \leq b \\ 1 & \text{if } r > b \end{cases}$$

Herrera and Herrera-Viedma (Herrera, Herrera-Viedma 2000) pointed out some examples of non-decreasing proportional fuzzy linguistic quantifiers in the form of *quantifier* (a,b): “most” (0.3,0.8), “at least half” (0,0.5) and “as many as possible” (0.5,1).

At the beginning of this chapter the two fundamental properties that any aggregation operator should satisfy were pointed out:

Property 1 (monotonicity): Assume F is an OWA operator. Let $A = [a_1, \dots, a_m]$ be an ordered argument vector. If $B = [b_1, \dots, b_m]$ is a second ordered argument vector such that for each j , $a_j \geq b_j$, then $F(A) \geq F(B)$.

Proof: Since $F(A) = W \cdot A$ and $F(B) = W \cdot B$ the result follows directly from the property $a_j \geq b_j$. \square

Property 2 (commutativity): If F is an OWA operator, then $F(a_1, a_2, \dots, a_m) = F(a'_1, a'_2, \dots, a'_m)$ where $(a'_1, a'_2, \dots, a'_m)$ is any permutation of the elements in (a_1, a_2, \dots, a_m) .

Proof: If B and B' are the ordered argument vectors of (a_1, a_2, \dots, a_n) and $(a'_1, a'_2, \dots, a'_m)$ respectively, then $B = B'$. Hence $F(B) = F(B')$. \square

3.2.1 Linguistic OWA operator

As commented before in Chapter 2, there are times in which preference values are more realistically represented using linguistic terms rather than using an exact numerical value. In that case, a scale of linguistic labels such as “Low”, “Medium” and “High”, can be used to express preference values over alternatives or over the values of the attributes that form those alternatives. The Linguistic OWA operator (Herrera et al. 1996) emerged as an aggregation tool in decision making processes where linguistic preferences are involved.

The LOWA operator is based on the ordered weighted averaging (OWA) operator defined by Yager (Yager 1988), and on the convex combination of linguistic labels defined by Delgado et al. (Delgado et al. 1993).

Definition 3.3. Let $\{a_1, \dots, a_m\}$ be a set of labels to aggregate. The LOWA operator ϕ is defined as

$$\begin{aligned} \phi(a_1, \dots, a_m) = W \cdot B^T = C^m \{w_k, b_k, k = 1, \dots, m\} = \\ w_1 \otimes b_1 \oplus (1 - w_1) \otimes C^{m-1} \{\beta_h, b_h, h = 2, \dots, m\}, \end{aligned} \quad (3.1)$$

where $W = [w_1, \dots, w_m]$, is a weighting vector, such that $w_i \in [0, 1]$ and $\sum_i w_i = 1$, $\beta_h = w_h / \sum_2^m w_k$, $h = 2, \dots, m$ and B is the associated ordered label vector. Each element $b_i \in B$ is the i -th largest label in the collection a_1, \dots, a_m . C^m is the convex combination operator of m labels and if $m = 2$ then it is defined as

$$C^2\{w_i, b_i, i = 1, 2\} = w_1 \otimes s_j \oplus (1 - w_1) \otimes s_i = s_k, \quad s_j, s_i \in S \quad (j \geq i)$$

such that $k = \min\{T, i + \text{round}(w_1 \cdot (j - i))\}$, where round is the usual round operation, $b_1 = s_j, b_2 = s_i$, and the set of linguistic labels $S = \{s_i\}, i \in \{0, \dots, T\}$.

If $w_j = 1$ and $w_i = 0 \forall i, i \neq j$, then the convex combination is defined as $C^m\{w_i, b_i, i = 1, \dots, m\} = b_j$.

The LOWA operator has some properties of the OWA operators investigated by Yager in (Yager 1988), such as monotonicity, commutativity, and the property to be an “orand” operator.

3.2.2 Induced OWA operator

Once the weights have been established, the aggregation policy is fully determined because the order of vector B in the OWA operator is based only on the value of the arguments a_j . However, as shown by Yager and Filev (Yager, Filev 1999), by allowing other orderings for the arguments we can obtain a more general aggregation operator: the Induced OWA

(IOWA). This generalization takes into account the ordering that an additional variable (u) induces in the set of values to be aggregated.

Definition 3.4. The IOWA operator is defined as follows (Yager, Filev 1999):

$$IOWA_W(\langle u_1, a_1 \rangle, \langle u_2, a_2 \rangle, \dots, \langle u_m, a_m \rangle) = W \cdot B_u = \sum_{j=1}^m w_j b_j, \quad (3.2)$$

where $W = (w_1, \dots, w_m)$ is the usual weighting vector that defines the aggregation policy of the OWA operator, with $w_i \in [0,1]$, $\sum w_i = 1$. The ordered argument vector B_u is obtained by taking b_j as the a_i value of the pair $\langle u_i, a_i \rangle$ which has the j -th largest u_i value. Yager refers to u as the order inducing variable and a as the argument variable.

It is important to note that the only requirement for the u variable is that it must be drawn from a space in which there is some linear ordering. This allows different kinds of criteria to be used for the order inducing variables. An important aspect of the IOWA operator is the fact that the order induced by the variable u can produce ties in some arguments. In this case, the relative order of two arguments a_i and a_j with $u_i = u_j$ is relevant because they may correspond to different values, that is $a_i \neq a_j$. Many authors adopt the solution of replacing a_i and a_j with their arithmetic average $(a_i + a_j)/2$. Another mechanism for solving ties consists of including a secondary ordering criterion (Merigó, Casanovas 2010), as it is proposed further in this document.

The IOWA operator has the properties of monotonicity, idempotency, symmetry, homogeneity, shift-invariance, and duality (Beliakov, James 2011).

The semantics of the OWA operator is a generalization of the idea of averaging or summarizing the arguments. However, IOWA permits other kinds of aggregation of the argument variables, which can be modelled by choosing the appropriate order inducing variable. Since the introduction of the IOWA operator, several authors have proposed different ways of inducing the order. For example, Pasi and Yager (Pasi, Yager 2006) used the IOWA to define the majority opinion in group decision making, by inducing the order of the arguments on the basis of the similarity among one value and its neighbours. The combination of this ordering criterion with linguistic quantifiers allows to calculate the satisfaction of the proposition “*the satisfaction value of most of the criteria*” rather than “*most of the criteria have to be satisfied*” (which would be the result of classical OWA). So, IOWA can be used to model different aggregation semantics. Merigó and Casanovas have developed several applications of IOWA with uncertain information (Merigó, Casanovas 2011b) and with distance measures (Merigó, Casanovas 2011a). Wei et al. (Wei et al. 2010) and Xia and Xu (Xia, Xu 2012) have studied several extensions by using intuitionistic fuzzy sets and fuzzy numbers.

The main advantage of the IOWA operator over the OWA operator is that it can deal with complex reordering processes where the highest value is not the first one in the reordering. Therefore, the induced variables solve an important drawback of the OWA operator, which is exclusively based on a weighting policy. For example, a journal may determine an optimal

average number of pages per paper. Thus, by using the IOWA operator we can ensure that extremely long papers are not first in the reordering process because they are not optimal in this analysis. Other interesting examples can be found when analysing several key variables of the human body including temperature, calories and weight.

The main classes of models with induced aggregation are classified (Beliakov, James 2011) as:

- *standard auxiliary ordering*, where the inducing variable is an attribute associated with the input that is not considered in the actual aggregation process but that is informative about the object itself.
- *nearest-neighbour rules*, where the order inducing variable represents the similarity or distance among the aggregating elements.
- *best-yesterday models*, applied in models where it is necessary to predict the order based on previous observations.
- *aggregation of complex objects*, in which it is necessary to operate with compound objects, such as aggregating matrices, where the order is not directly defined and needs to be estimated with some additional measure.
- *group decision making*, an area in which it has been proposed that the consensus can be better achieved with inducing variables based on the support of each individual score.
- *multiple inducing variables*, where a priority order is established among more than one inducing variable.

3.3 Unbalanced LOWA operator

In LOWA, a set of equally-informative (*balanced*) terms is assumed. That is, the membership functions of the fuzzy terms are symmetric and the terms are uniformly distributed around a mid (*i.e.*, neutral) term. For example, when evaluating the users' satisfaction on some topic, linguistic labels like "very low", "low", "almost low", "medium", "high", "very high" and "perfect" can be used (Xu 2009).

However, recently many authors have noticed that there are some problems that require a more flexible definition of the set of linguistic terms (Herrera et al. 2008; Xu 2009). In some cases, fuzzy sets should be asymmetric or more positive than negative terms are needed (or *vice-versa*), leading to an *unbalanced* set of labels. An example is the set of terms used in grading evaluations $S = \{F, D, C, B, A\}$ (being D the neutral; A , B and C , positive marks, and F – fail – the unique negative value). Many other real-life applications also make use of unbalanced terms, such as (Xu 2009; Martínez et al. 2007; Kim et al. 2002).

Some extensions of well-known aggregation operators have been defined to deal with unbalanced sets of terms. The proposals of (Herrera et al. 2008) and (Cabrerizo et al. 2009) are based on the linguistic 2-tuple representation model, which attaches to each label a deviation value that permits to work with more precision. In (Xu 2009) the Uncertain Linguistic Weighted Average is proposed. It restricts to the case of sets of labels

where the absolute value of the deviation between the indices of two adjacent labels increases as the indices increase. An additional number is attached to each label, indicating its degree of separation from the mid-term. The aggregation operator uses those additional numbers to calculate the result.

Here it is assumed the classic fuzzy linguistic model that associates a fuzzy membership function to each linguistic value and operates in terms of those fuzzy sets. The LOWA operator defined in (Herrera et al. 1996) is the basis of the new operator, called ULOWA (Unbalanced Linguistic Ordered Weighted Average) which was defined in (Isern et al. 2010). LOWA has been chosen because it has been extensively used in many applications and it is computationally efficient. LOWA relies on a symbolic model of the set of linguistic terms, and the aggregation is done considering only the positions of the labels in the set of terms, but without taking into account any fuzzy set associated to the terms.

The aggregation procedure of ULOWA is the same than the one used in LOWA. However, in the convex combination of two labels, ULOWA exploits the knowledge given by the membership functions during the aggregation procedure. This approach, although it is focused on managing unbalanced label sets, can also be used with balanced label sets giving the same results than the LOWA operator.

3.3.1 Preliminaries

Linguistic values

Let us consider a set of linguistic labels $S = \{s_i, i \in \{0, \dots, T\}\}$. This set is defined as usual (Bonissone, Decker 1985; Herrera et al. 1996; Xu 2008) taking S as a finite and totally ordered term set on a reference domain $X = [0,1]$, with an odd cardinal, where one of the labels corresponds to the neutral value and the remaining terms are placed around it. The cardinality of the set must be small enough so as not to impose useless precision and rich enough in order to allow an appropriate discrimination level. The usual cardinality values are 7 or 9.

The semantics of the linguistic labels are given by a trapezoidal membership function $\mu: X \rightarrow [0,1]$ that is represented with a tuple $A = (a_1, a_2, a_3, a_4)$, where a_1, a_2, a_3 and a_4 are the points in the reference domain X which define the trapezoid (see Figure 8). Some special cases can be defined. If $a_1 = a_2$ and $a_3 = a_4$ then A corresponds to a crisp interval. If $a_2 = a_3$ the fuzzy set A is triangular. If $a_1 = a_2 = a_3 = a_4$, then A is called a crisp real number. This last case will be especially relevant for the aggregation method that is proposed further in this document.

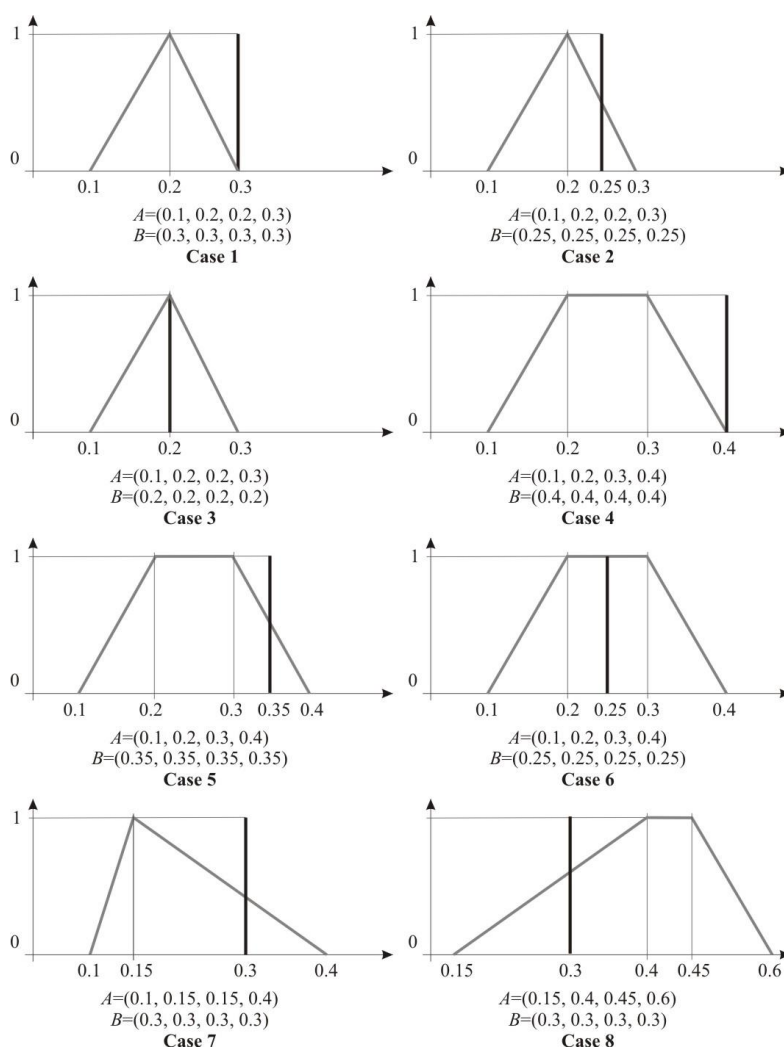


Figure 8. Combinations used to analyze the similarity function

Centre of gravity

The function to calculate the centre of gravity (COG) of a fuzzy term presented here is required to for the similarity function introduced further in this document.

Definition 3.5. Let us denote the centre of gravity (COG) of a trapezoidal fuzzy set A as $COG(A) = (x_A^*, y_A^*)$, which can be calculated as follows (Chen, Chen 2003):

$$y_A^* = \begin{cases} \frac{1}{6} \left(\frac{a_3 - a_2}{a_4 - a_1} + 2 \right) & , \text{ if } a_1 \neq a_4 \\ \frac{1}{2} & , \text{ if } a_1 = a_4 \end{cases} \quad (3.3)$$

$$x_A^* = \frac{y_A^* (a_3 + a_2) + (a_4 + a_1)(1 - y_A^*)}{2} \quad (3.4)$$

Similarity between fuzzy sets

The function to calculate the similarity between two fuzzy sets introduced here is a key element in the definition of the ULOWA operator included further in this document.

Several functions to measure the degree of similarity between two fuzzy sets have been defined. In this case, as it will be shown in the next section, a function that measures the similarity between a crisp number and a trapezoidal or triangular fuzzy set is needed. For this reason, we will consider the ones referenced in Table 1 (Yong et al. 2004; Chen 1996; Chen 2006; Chen, Chen 2003; Hsieh, Chen 1999). These functions are based on the different properties of the membership function of the fuzzy set, such as the centre of gravity, the radius of gyration, or the geometric mean. For any similarity function, the following three properties should be satisfied (Chen, Chen 2003). Given two fuzzy sets A and B :

1. A and B are identical if and only if $Sim(A, B) = 1$
2. $Sim(A, B) = Sim(B, A)$
3. If $A = (a, a, a, a)$ and $B = (b, b, b, b)$ denote two crisp numbers in $[0,1]$, then $Sim(A, B) = 1 - |a - b|$.

Figure 8 shows different combinations where there is a crisp number and a fuzzy set (triangular or trapezoidal). Table 1 gives the results obtained with different similarity functions in those 8 cases. For this purpose, some relations between the similarity values obtained in each of those cases should be satisfied. In the sequences defined by the cases 1, 2, 3, and 4, 5, 6, the crisp value has different degrees of overlapping with the fuzzy set. In those cases, the similarity value should increase as the crisp value gets inside the fuzzy set. Moreover, the similarity in case 3 should be greater than in cases 4 and 5, and even 6.

Table 1. Comparison of similarity functions between fuzzy sets

Case	Hsieh and Chen (Hsieh, Chen 1999)	Chen (Chen 1996)	Chen and Chen (Chen, Chen 2003)	Yong (Yong et al. 2004)	Chen (Chen 2006)	(Chen 2006) Modified
1	0.9091	0.9000	0.5400	0.5854	0.5991	0.8987
2	0.9524	0.9250	0.5858	0.6344	0.6163	0.9245
3	1.0000	0.9500	0.6333	0.6584	0.6329	0.9494
4	0.8696	0.8500	0.5619	0.6387	0.6585	0.8466
5	0.9091	0.8750	0.6125	0.6705	0.6791	0.8731
6	1.0000	0.9000	0.7000	0.7656	0.6995	0.8993
7	0.8955	0.8500	0.5194	0.5821	0.5664	0.8497
8	0.9023	0.8250	0.5268	0.5620	0.5794	0.8234

“” Anomalous results are highlighted

Existing similarity measures do not fulfil these conditions (marked in Table 1 as anomalous results). In the case of Hsieh and Chen (Hsieh, Chen 1999), the cases 1 and 5 cannot be distinguished. The Chen (Chen 1996) function cannot distinguish the cases 1 and 6. The rest of the functions give a lower similarity in the case 3 when compared with cases 4, 5 or 6.

Therefore, the similarity function between two fuzzy sets A and B defined by Chen (Chen 2006), which was proven to be more adequate than the rest (details in (Chen 2006)), has been chosen and conveniently adapted. In particular, the scale factor designed to compare two general fuzzy numbers with different heights has been avoided provided that, in the

scenario of the work conducted in this document, the height of the membership function is always 1. The new function is given in Eq.(3.5). The results obtained with this similarity function are given in the last column of Table 1.

$$Sim(A, B) = \sqrt[4]{\prod_{i=1}^4 (2 - |a_i - b_i|)} - 1 \quad (3.5)$$

3.3.2 Definition of the ULOWA operator

The new operator, called ULOWA (Unbalanced Linguistic Ordered Weighted Average), is defined on the basis of the LOWA operator. In fact, the definition is the same until we have to make the aggregation of two labels ($m=2$). The convex combination of two terms $b_1=s_j$ and $b_2=s_i$, with $s_j, s_i \in S(j \geq i)$ is calculated taking into account the membership functions of labels s_j and s_i :

$$C^2\{w_i, b_i, i=1,2\} = w_1 \otimes s_j \oplus (1-w_1) \otimes s_i = s_k \text{ such that} \quad (3.6)$$

$$k = \arg \max_{i \leq p \leq j} \{Sim(s_p, \delta)\}$$

where δ is an intermediate crisp number defined as $\delta = (x_k, x_k, x_k, x_k)$ with $x_k = x_{s_i}^* + w_1(x_{s_j}^* - x_{s_i}^*)$. Note that the COG of labels s_j and s_i (calculated using Eq.(3.4)), and the similarity function for two fuzzy sets (calculated using Eq.(3.5)) described above, are used.

Figure 9 illustrates the aggregation procedure of the two extreme labels (VL and P) of an unbalanced set with 7 terms with a “mean” policy (which is a linguistic quantifier where $a = b = 0.5$). The figure shows the COGs of both labels and their intermediate crisp number δ used to find the result of the aggregation. After the application of the similarity function, Eq.(3.5), the resulting label is the neutral term M . Using the LOWA operator, the result would have been AH , which is not the intuitive expected result of the mean average of these two opposite evaluations, considering the meaning of the terms.

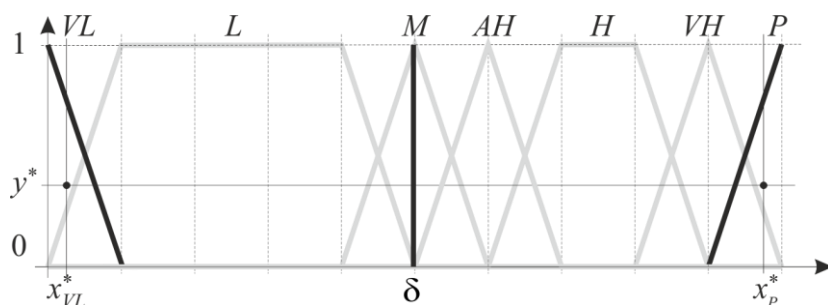


Figure 9. Graphical example of ULOWA when aggregating two labels (VL and P) with a mean average

3.3.3 Properties of the ULOWA operator

As stated in (Yager 1988), an OWA operator should satisfy the following properties: *monotonicity*, *commutativity*, *idempotency*, and *to be an “orand” operator*.

Property 1: *The ULOWA operator ϕ is increasing monotonous with respect to the argument values, in the following sense:*

If $A = [a_1, \dots, a_m]$ and $B = [b_1, \dots, b_m]$ are two ordered argument vectors, such that $\forall j, a_j \geq b_j$ then $\phi(A) \geq \phi(B)$.

Proof: By induction over the number of arguments to aggregate.

a) For $m=2$. Let $x_{s_j}^*, x_{s_i}^*, x_{s_q}^*, x_{s_p}^*$ be the x coordinates of the centre of gravity of the ordered labels s_j, s_i, s_q, s_p corresponding to a_1, a_2, b_1, b_2 , respectively, δ_{ji} the intermediate point between s_j and s_i , and δ_{qp} the intermediate point between s_q and s_p . In this case, $x_{s_j}^* \geq x_{s_q}^*$ and $x_{s_i}^* \geq x_{s_p}^*$; therefore, given any $w_1 \in [0,1]$, $x_{s_j}^* \cdot w_1 \geq x_{s_q}^* \cdot w_1$ and $x_{s_i}^* \cdot (1 - w_1) \geq x_{s_p}^* \cdot (1 - w_1)$, and thus $x_{s_j}^* \cdot w_1 + x_{s_i}^* \cdot (1 - w_1) \geq x_{s_q}^* \cdot w_1 + x_{s_p}^* \cdot (1 - w_1)$.

From this expression it can be derived that $x_{s_i}^* + w_1 \cdot (x_{s_j}^* - x_{s_i}^*) \geq x_{s_p}^* + w_1 \cdot (x_{s_q}^* - x_{s_p}^*)$.

Knowing that $\delta_{ji} = (x_k, x_k, x_k, x_k)$ with $x_k = x_{s_i}^* + w_1(x_{s_j}^* - x_{s_i}^*)$ and $\delta_{qp} = (x_r, x_r, x_r, x_r)$ with $x_r = x_{s_p}^* + w_1(x_{s_q}^* - x_{s_p}^*)$, we can deduce $x_k \geq x_r$.

Given a fuzzy number A , its similarities with the previous values using the similarity function shown in Eq.(3.5) are

$$Sim(A, \delta_{ji}) = \left[\sqrt[4]{\prod_{i=1}^4 (2 - |a_i - x_k|)} - 1 \right], \text{ and}$$

$$Sim(A, \delta_{qp}) = \left[\sqrt[4]{\prod_{i=1}^4 (2 - |a_i - x_r|)} - 1 \right].$$

Notice that the relation between $Sim(A, \delta_{ji})$ and $Sim(A, \delta_{qp})$ only depends on the distances between the values that define the fuzzy number A and x_k and x_r respectively. Thus, *the smaller the distances* ($|a_i - x_k|$ and $|a_i - x_r|$), the bigger the similarities are.

So if we calculate a s_k and a s_r that

$$k = \operatorname{argmax}_{i \leq h \leq j} \{Sim(s_h, \delta_{ji})\}, s_k = \phi(a_1, a_2), \text{ and}$$

$$r = \operatorname{argmax}_{p \leq n \leq q} \{Sim(s_n, \delta_{qp})\}, s_r = \phi(b_1, b_2),$$

then s_k is closer (or equal) to δ_{ji} than to δ_{qp} and s_r is closer (or equal) to δ_{qp} than to δ_{ji} , so if $x_k \geq x_r$, then $s_k \geq s_r$ and this proves that $\phi(a_1, a_2) \geq \phi(b_1, b_2)$.

b) Suppose that for $m - 1$, $\phi(a_1, \dots, a_{m-1}) \geq \phi(b_1, \dots, b_{m-1})$.

Consider the aggregation of m values on A and B :

$$\phi(a_1, \dots, a_m) = w_1 \otimes a_1 \oplus (1 - w_1) \otimes C^{m-1}\{\beta_h, a_h, h = 2, \dots, m\},$$

$$\text{with } C^{m-1}\{\beta_h, a_h, h = 2, \dots, m\} = \phi(a_2, \dots, a_m) = s_j, \text{ and}$$

$$\phi(b_1, \dots, b_m) = w_1 \otimes b_1 \oplus (1 - w_1) \otimes C^{m-1}\{\beta_h, b_h, h = 2, \dots, m\},$$

$$\text{with } C^{m-1}\{\beta_h, b_h, h = 2, \dots, m\} = \phi(b_2, \dots, b_m) = s_i.$$

By induction hypothesis $\phi(a_1, a_2, \dots, a_m) = \phi(a_1, s_j)$ and $\phi(b_1, b_2, \dots, b_m) = \phi(b_1, s_i)$, which reduces it to the case $m = 2$.

As $s_j \geq s_i$, $a_1 \geq b_1$, $a_1 \geq s_j$ and $b_1 \geq s_i$, it can be concluded that $\phi(a_1, s_j) \geq \phi(b_1, s_i)$. Therefore $\phi(a_1, a_2, \dots, a_m) \geq \phi(b_1, b_2, \dots, b_m)$. \square

Property 2: *The ULOWA operator ϕ is commutative, i.e., $\phi(a_1, \dots, a_m) = \phi(a'_1, \dots, a'_m)$ where (a'_1, \dots, a'_m) is any permutation of the elements in (a_1, \dots, a_m) .*

Proof: The ULOWA operator makes an ordered weighted average of the arguments. If A and A' are the ordered argument vectors of (a_1, \dots, a_m) and (a'_1, \dots, a'_m) respectively, then $A = A'$. So, $\phi(A) = \phi(A')$. \square

Property 3: *The ULOWA operator ϕ is idempotent in the sense that if $\forall j, a_j = a$, then $\phi(a_1, \dots, a_m) = a$.*

Proof: Following the definition of the ULOWA operator, the final step consists in $\phi(a_{m-1}, a_m) = s_k$ where

$$k = \operatorname{argmax}_{i \leq p \leq j} \left\{ \operatorname{Sim} \left(s_p, (x_k, x_k, x_k, x_k) \right) \right\}.$$

In this case, $p = i = j$ and $s_k = s_i = a$.

Recursively, it is obtained that $\phi(a_1, \dots, a_m) = a$. \square

Property 4: *The ULOWA operator ϕ is an “orand” operator. That is, for any weighting vector W and ordered labels $A = [a_1, \dots, a_m]$ ($a_1 \geq a_2 \geq \dots \geq a_m$), then:*

$$\min(a_1, \dots, a_m) \leq \phi(a_1, \dots, a_m) \leq \max(a_1, \dots, a_m).$$

Proof: Being $s_j, s_i \in S(j \geq i)$, the convex combination of these two terms has been defined as $s_k = w_1 \otimes s_j \oplus (1 - w_1) \otimes s_i$ (Delgado et al. 1993). According to Eq.(3.6) $k = \operatorname{argmax}_{i \leq p \leq j} \{Sim(s_p, \delta)\}$. Therefore, $i \leq k \leq j$, that is, the resulting label from the combination of two labels ($\phi(s_i, s_j)$) is s_k with $i \leq k \leq j$, proving that $a_m \leq \phi(a_1, \dots, a_m) \leq a_1$. \square

3.3.4 Examples

In this section two examples of the ULOWA operator are analysed. The first is a numerical example, and the second illustrates the behaviour of the operator when the fuzzy membership functions of the terms change.

Numerical example

Let us take five labels to aggregate using the ULOWA operator. The granularity of the fuzzy set is 7 and the definition of the membership functions is shown in Figure 10b. The weighting vector is $W = (0.2, 0.2, 0.2, 0.2, 0.2)$, which corresponds to the “mean” average. The ordered set of linguistic labels to aggregate is P, H, M, L and VL .

The ULOWA aggregation begins following (3.1):

$$\begin{aligned} \phi(P, H, M, L, VL) &= W \cdot B^T = C^5\{w_k, b_k, k = 1, \dots, 5\} = \\ &= 0.2 \otimes P \oplus (1 - 0.2) \otimes C^4\{\beta_h, b_h, h = 2, \dots, 5\}. \end{aligned}$$

Now, C^4 needs to be evaluated as: $C^4\{\beta_h, b_h, h = 2, \dots, 5\} =$

$$= \left(\begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix} \otimes H \oplus \left(1 - \frac{0.2}{0.8}\right) \otimes C^3\{\beta_h, b_h, h = 3, \dots, 5\} \right),$$

where $C^3\{\beta_h, b_h, h = 3, \dots, 5\} =$

$$= \left(\begin{pmatrix} 0.2 \\ 0.6 \end{pmatrix} \otimes M \oplus \left(1 - \frac{0.2}{0.6}\right) \otimes C^2\{\beta_h, b_h, h = 4, 5\} \right).$$

In the last step, C^2 is calculated as follows:

$$C^2\{\beta_h, b_h, h = 4, 5\} = \left(\begin{pmatrix} 0.2 \\ 0.4 \end{pmatrix} \otimes L \oplus \left(1 - \frac{0.2}{0.4}\right) \otimes VL \right) = s_k.$$

Now, the two terms to aggregate are $s_j = L$ and $s_i = VL$. The corresponding centers of gravity (see Eq.(3.4)) of L and VL are $COG(L) = (0.23, 0.33)$ and $COG(VL) = (0.03, 0.33)$. The intermediate point δ is calculated as follows:

$$x_k = x_{VL}^* + w_1(x_L^* - x_{VL}^*) = 0.03 + 0.5 \cdot (0.23 - 0.03) = 0.13.$$

The similarities of VL and L with $\delta = (0.13, 0.13, 0.13, 0.13)$ are 0.89 and 0.83, respectively (using Eq.(3.5)). According to the similarity function that has been defined in Eq.(3.6), the label with maximum similarity is $s_k = VL$, because it is quite near the point 0.13 and has a more precise meaning than L , which is more vague (this is penalized in Eq.(3.6)).

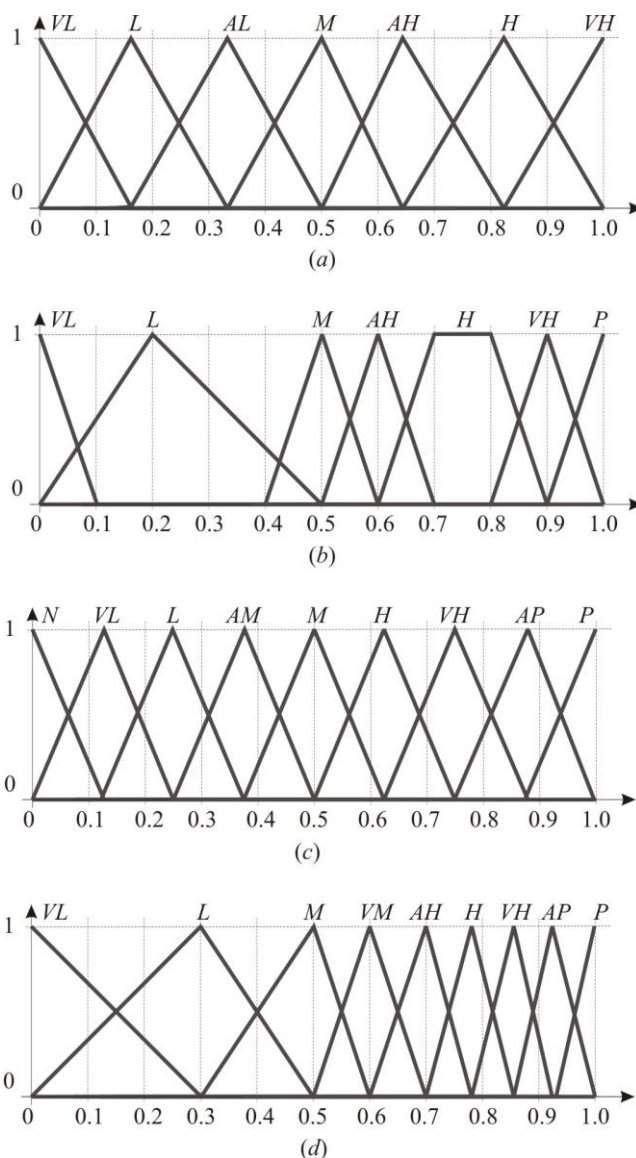


Figure 10. Fuzzy term sets with 7 and 9 labels: (a) and (c) balanced; (b) and (d) unbalanced.

Then, the same approach continues to obtain:

$$C^3\{\beta_h, b_h, h = 3, \dots, 5\} = \left(\left(\frac{0.2}{0.6} \right) \otimes M \oplus \left(1 - \frac{0.2}{0.6} \right) \otimes VL \right) = L.$$

The intermediate point is $\delta = (0.19, 0.19, 0.19, 0.19)$. The similarities of VL , L and M with δ are 0.83, 0.86 and 0.68 respectively. Notice that, in this case, the right side of the membership function of L moves its centre of gravity to the right, increasing its similarity with δ . So, the result is L .

The next step is:

$$C^4\{\beta_h, b_h, h = 2, \dots, 5\} = (0.25 \otimes H \oplus 0.75 \otimes L) = M.$$

In this step, $\delta = (0.36, 0.36, 0.36, 0.36)$. The similarities of δ with L , M , AH , and H are 0.79, 0.86, 0.76, and 0.60 respectively. Thus, the most similar label to δ is M .

Finally, the last step gives the final result:

$$\phi(P, H, L, L, VL) = W \cdot B^T =$$

$$C^5\{w_k, b_k, k = 1, \dots, 5\} = 0.2 \otimes P \oplus (1 - 0.2) \otimes M = AH.$$

The resulting intermediate point is $\delta = (0.59, 0.59, 0.59, 0.59)$. The similarities of δ with M, AH, H, VH and P are 0.90, 0.94, 0.83, 0.69, and 0.61 respectively. The resulting label of the aggregation is AH .

Qualitative analysis

In this section an example to illustrate the influence of the membership function of the terms in the result of the ULOWA aggregation is shown. Figure 11 shows three cases aggregating the same labels (VL and AH) with the mean average policy. In each case, we have a different fuzzy membership function for the term L . Notice that in case 1 the result of the aggregation given by the ULOWA operator is M , whereas in cases 2 and 3 the result is L . This difference is due to the influence of the membership function of L .

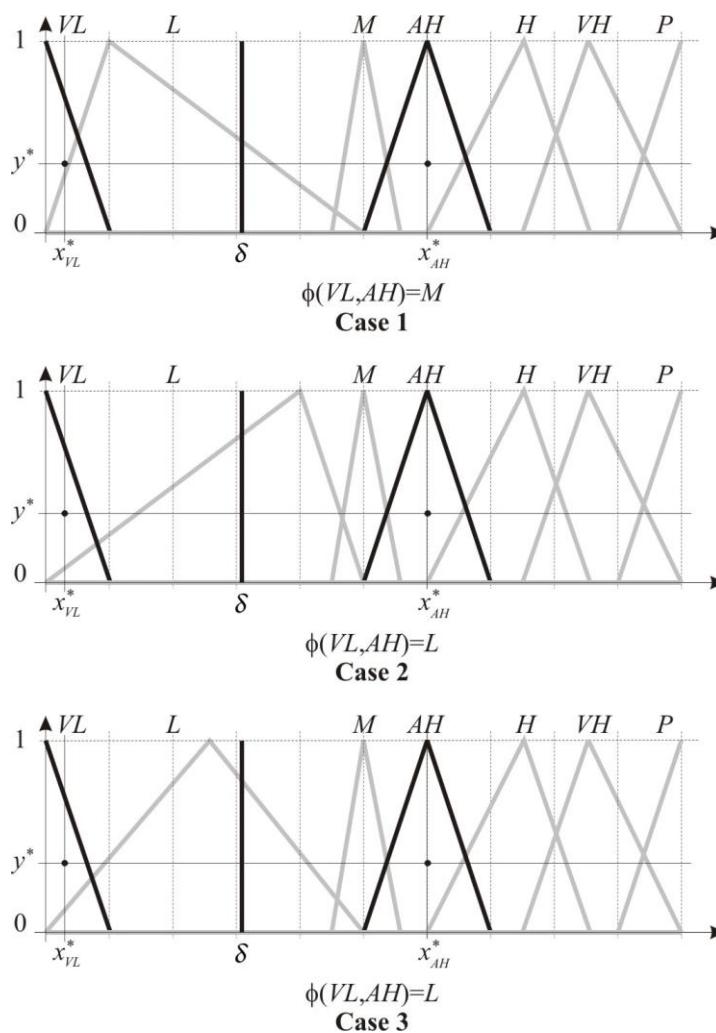


Figure 11. Examples of ULOWA when aggregating VL and AH with an average weight.

In the first case, the closest label to δ (the intermediate point between the centres of gravity of both labels) is M , because the label L is asymmetric and its centre of gravity is located in the left hand side, that is, it has a more negative meaning than the term L defined in cases 2 and 3. On the other two cases, we obtain the label L because the point of the label with maximum membership is close to δ .

This example shows that ULOWA is sensible to the definition of the semantics of the fuzzy terms, which is especially appropriate when we want to deal with unbalanced sets of terms.

3.3.5 Comparison with other linguistic operators

LOWA versus ULOWA

Dealing with linguistic labels is usually made with symbolic algorithms that use the order of terms and negation functions. These approaches do not use the membership functions associated to the labels and, with unbalanced label sets, they exhibit undesirable (or non-logical) results.

As previously stated, the LOWA operator deals with linguistic information taking into account any balanced distribution of terms (Herrera et al. 1996). The operations are defined at a symbolic level, considering only the position of terms in the set S . For this reason it cannot be applied in the case of unbalanced linguistic term sets. On the contrary, ULOWA has been specially designed for the unbalanced case. However, it should behave correctly also with balanced sets of terms.

Table 2 presents the results obtained with LOWA and ULOWA in different situations. This table shows the aggregation of different sequences of labels, with different policies (*e.g.*, mean, as many as possible), as well as taking into account different fuzzy sets (shown in Figure 10). When applying ULOWA on balanced sets (term sets a and c in Figure 10), both operators exhibit a very similar behaviour, with the exception of entry 18. This is due to the fact that when dealing with boundary labels on the aggregations (such as VL or VH with 7-label sets), their membership functions are different from the rest, and their COGs are closer to the adjacent label than the rest. Since ULOWA uses the information of the centre of gravity, the result obtained with ULOWA when the boundary labels are aggregated can be different than the result with the LOWA operator.

When dealing with unbalanced sets of terms (sets b and d on Figure 10), the behaviour of both operators is different. The influence of membership functions, as shown in the previous section, permits to obtain more logical results. For instance, entry 11 aggregates the best (P) and the worst (VL) options with nine labels (set d). The LOWA operator just returns their intermediate label AH , but this result is not logical attending to the used fuzzy set. In this case, ULOWA returns M , the neutral term.

Table 2. Comparison of LOWA and ULOWA operators

Entry	Set ^a	Labels to aggregate	Policy ^b	LOWA	ULOWA
1	(a)	VL VH	AV	M	M
2	(a)	VL H	AV	M	M
3	(a)	VL AH	AV	AL	AL
4	(a)	VL M	AV	AL	AL
5	(a)	L M	AV	AL	AL
6	(b)	VL P	AV	AH	M
7	(b)	VL VH	AV	AH	M
8	(b)	VL H	AV	M	M
9	(b)	VL AH	AV	M	M
10	(b)	VL M	AV	L	L
11	(d)	VL P	AV	AH	M
12	(d)	VL AP	AV	AH	M
13	(d)	L AP	AV	AH	VM
14	(d)	M AP	AV	H	AH
15	(a)	VL L AL AH VH	ALH	AH	AH
16	(a)	VL L AL AH VH	AV	M	M
17	(a)	VL L AL AH VH	Most	AL	AL
18	(a)	VL L AL AH VH	AMAP	L	VL
19	(a)	VL VL L VH VH	ALH	H	H
20	(a)	VL VL L VH VH	AV	M	M
21	(a)	VL VL L VH VH	Most	AL	AL
22	(a)	VL VL L VH VH	AMAP	VL	VL
23	(b)	VL L M H P	ALH	H	H
24	(b)	VL L M H P	AV	AH	AH
25	(b)	VL L M H P	Most	M	M
26	(b)	VL L M H P	AMAP	L	VL
27	(b)	VL VL M P P	ALH	VH	P
28	(b)	VL VL M P P	AV	AH	AH
29	(b)	VL VL M P P	Most	M	M
30	(b)	VL VL M P P	AMAP	VL	VL
31	(b)	L M M H P P	ALH	VH	VH
32	(b)	L M M H P P	AV	H	AH
33	(b)	L M M H P P	Most	AH	AH
34	(b)	L M M H P P	AMAP	L	M
35	(d)	VL VL M P P	ALH	AP	VH
36	(d)	VL VL M P P	AV	AH	VH
37	(d)	VL VL M P P	Most	M	M
38	(d)	VL VL M P P	AMAP	VL	VL
39	(c)	VL L L VH P P	ALH	AP	AP
40	(c)	VL L L VH P P	AV	M	M
41	(c)	VL L L VH P P	Most	AM	AM
42	(c)	VL L L VH P P	AMAP	VL	VL
43	(d)	L M M VH P P	ALH	AP	AP
44	(d)	L M M VH P P	AV	AH	AH
45	(d)	L M M VH P P	Most	VM	VM
46	(d)	L M M VH P P	AMAP	L	M

^a Fuzzy sets depicted in Figure 10

^b Decision-maker policies: Most, ALH(“At least half”), AMAP (“As many as possible”) and AV(“Average”).

The influence of the decision-makers policy is also compared in Table 2. The policies are sorted according to its corresponding *orness* (Yager 1988): *at least half* (0.8), *average* (0.5), *most* (0.45), and *as many as possible* (0.25). With those different aggregation weights and an unbalanced set of terms, the LOWA and ULOWA give 12 different results out of 29 cases (entries from 6 to 14, from 23 to 38 and from 43 to 46). For example, for entry 46 that aggregates (L, M, M, VH, P, P) with “as many as possible”, the result of ULOWA is M instead of L (given by LOWA), because it is able to recognise that M is the neutral value and there are 3 very high evaluations.

ULOWA vs other unbalanced aggregation operators

Recently, there have been some studies proposing the use of unbalanced sets of terms in which some unbalanced aggregation operators have appeared. In (Herrera et al. 2008) a new model for representing unbalanced linguistic information is defined. It is based on the *2-tuple* fuzzy model for semantically interpreting the terms, as well as on a hierarchy of balanced linguistic term sets with different granularity. With the combination of parts of this linguistic hierarchy, one can define the unbalanced linguistic term set by means of the appropriate transformation functions. Then, any of the aggregation operators of linguistic 2-tuples can be used (Herrera, Martínez 2000).

In the 2-tuple model, terms are represented by a pair (s, α) , where s is the linguistic label and α is a number that represents the translation of symbols into a real scale. This model is able to deal with unbalanced linguistic variables by means of hierarchical linguistic contexts. A linguistic hierarchy (Herrera, Martínez 2001) is a set of levels, where each level is a balanced linguistic term set with a granularity different from that of the remaining levels in the hierarchy. Each level belonging to a linguistic hierarchy is denoted as $l(t, n(t))$, where t is the number that indicates the level of the hierarchy, and $n(t)$ indicates the granularity of the linguistic term set of that level. To build a linguistic hierarchy, the authors propose that the linguistic set of terms for level $t + 1$ is obtained from its predecessor using the expression $L(t, n(t)) \rightarrow L(t + 1, 2 \cdot n(t) - 1)$.

Working with terms from different levels of the hierarchy necessitates the use of transformation functions to translate linguistic terms from one level to another. Considering that $LH = U_t l(t, n(t))$ is a linguistic hierarchy whose linguistic term sets are denoted as $S^{n(t)} = \{s_0^{n(t)}, \dots, s_{n(t)-1}^{n(t)}\}$, a transformation function from a linguistic label in level t to a label in level t' is defined as:

$$TF_t^{t'}(s_i^{n(t)}, \alpha^{n(t)}) = \Delta \left(\frac{\Delta^{-1}(s_i^{n(t)}, \alpha^{n(t)}) \cdot (n(t') - 1)}{n(t) - 1} \right) \quad (3.7)$$

where $\Delta(\beta) = (s_{\text{round}(\beta)}, \beta - \text{round}(\beta))$ and $\Delta^{-1}(s_i, \alpha) = i + \alpha$.

Herrera-Viedma *et al.* (Herrera-Viedma et al. 2011) define the LOWA_{un} and ILOWA_{un} operators on the basis of these transformation functions in order to make an aggregation (and an induced aggregation) of unbalanced linguistic terms.

To give an illustrative example, Figure 12a shows a 3-level linguistic hierarchy. The parts in red are used to construct the unbalanced term set depicted in Figure 12b.

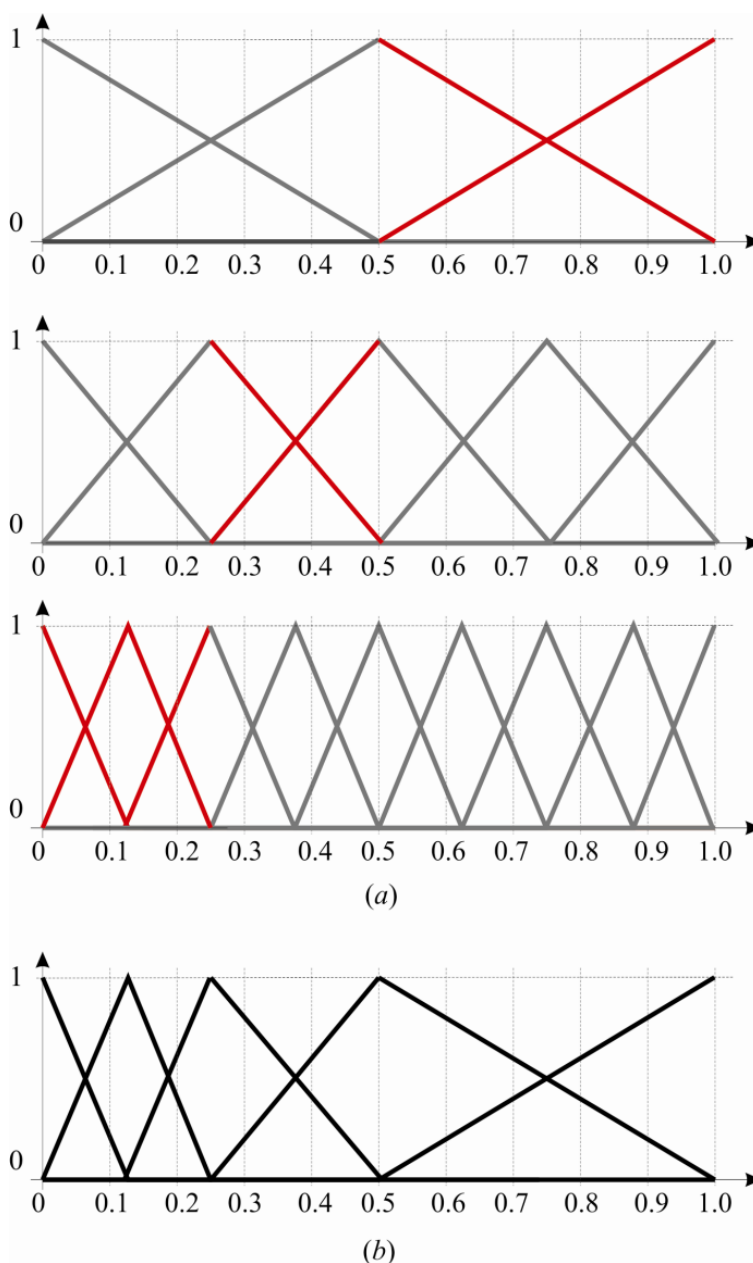


Figure 12. Unbalanced term set with 5 linguistic labels (b) obtained from a linguistic hierarchy of 3 levels (a)

As can be seen in this example, the set of labels used in each of the levels of the hierarchy is balanced and uniformly distributed but, by taking some pieces of each level and putting them together, it is possible to model an unbalanced term set. The main drawbacks of this approach are that it is quite complex to define an appropriate set of levels and the number of levels (and labels) to consider can be quite large. In the example shown above, there are a total of 17 labels in the 3 levels of the hierarchy, whereas the unbalanced term set to be modelled only has 5 labels. Moreover, it is assumed that the fuzzy sets are subsumed in the hierarchy of labels (usually with a high number of labels). This assumption is complex to fulfil in some unbalanced and asymmetric sets such as those mentioned in (Garibaldi, Ifeachor 2000) and (Hong, Chen 1999). On the contrary, the ULOWA operator uses standard membership functions to define each of the linguistic terms, requiring only an order between the terms and the complete coverage of the reference domain. Moreover, the approach

presented in this document supports a wide range of situations, working with any combination of triangular and trapezoidal functions, with different degrees of overlapping, as shown in Figure 10 and in Figure 11.

Another prominent proposal for modelling unbalanced term sets is given by Xu (Xu 2009). He argues that, when defining an unbalanced term set, the absolute value of the deviation between the indices of two adjoining linguistic labels should increase as the indices of the linguistic labels steadily increase (the term in the centre has index 0, so there are terms with positive and negative indices). Following this idea, he proposes defining a term set with $2t-1$ labels in the following way:

$$S^t = \left\{ s_{\beta}^t \mid \beta = 1-t, \frac{2}{3}(2-t), \frac{2}{4}(3-t), \dots, 0, \dots, \frac{2}{4}(t-3), \frac{2}{3}(t-2), t-1 \right\} \quad (3.8)$$

For example, Figure 13 shows an unbalanced term set with nine labels.

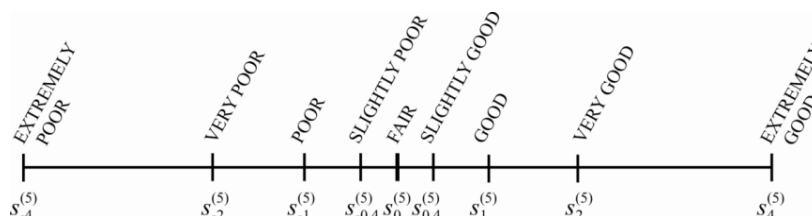


Figure 13. A set of nine linguistic labels (from (Xu 2009))

It can be seen that this way of defining the unbalanced term sets is very rigid. It is only possible to model those situations in which the labels in the middle are very precise and the labels in the extremes have a wider range. Moreover, the labels are symmetrically located with respect to the centre of the domain, so it is not possible to have more positive than negative labels. This strict definition of the term sets allows Xu to define simple functions that permit terms in one set to be transformed into terms in another set. These transformations are meant to be used when different experts have used different term sets to evaluate a set of alternatives. Each label is basically represented by a point in the domain, rather than by a fuzzy set.

Another important difference between our work and Xu's proposal (Xu 2009) is that he deals with uncertain linguistic values, in which each expert's assessment of each criteria for each alternative is represented by an interval of labels, rather than by a single value (e.g. an expert could say that the quality of an object's attribute is "between Good and Very Good"). Thus, the expert's opinions regarding a set of alternatives are represented by a matrix of intervals of labels, rather than by a matrix of labels.

One of the main aims of our work was to devise a method for working with any unbalanced set of terms, without any restriction on the definition of the fuzzy set associated to each term (as long as a fuzzy partition is obtained). Xu's proposal (Xu 2009) does not provide this flexibility, since the term sets are very precisely defined and have to be symmetrically located with respect to the centre of the domain. Moreover, it only considers situations in which precise labels are required in the centre and imprecise labels in the extremes. The studies by Herrera-Viedma *et al.* (Herrera et al. 2008; Herrera-Viedma et al. 2011) allow some unbalanced sets to be modelled provided that the labels are composed by taking pieces

from each level of the hierarchy. In contrast, our approach permits the direct definition of the unbalanced term set that best fits the needs of the application, choosing any fuzzy set for each label. The price to be paid for this flexibility is that the aggregation operator must operate on fuzzy sets.

As a summary, the compared methods do not use membership functions when combining labels, and they only take into account their position.

3.3.6 ULOWA Conclusions

The reader can find some linguistic approaches to aggregate information, but they do not operate with the fuzzy membership functions of the terms. For the development of the ULOWA operator, the focus has been put on the LOWA operator, which works at a purely symbolic level, taking into account only the position of terms in the set, assuming a set of balanced terms. This proposal extends the LOWA definition, including the information provided by fuzzy sets. This is used to decide in each step of the aggregation which label is the most appropriate result according to its semantics. The proposed algorithm is able to work with both balanced and unbalanced fuzzy sets.

The ULOWA operator uses a similarity function to compare fuzzy labels and decide the resulting label of each aggregation step. Some existing similarity functions defined for fuzzy sets do not fit its requirements, so a new one had to be proposed, based on previous work by Chen (Chen 2006). Combining these elements, it has been shown how ULOWA is sensitive to the fuzzy membership functions of the labels. This fact gives the user more freedom when defining the sets according to his/her requirements. It has been illustrated how this operator works and how it reacts to the change of one of the membership functions.

As seen on the evaluation, in some extreme cases, the results can be slightly different to the ones obtained with LOWA when using a balanced set of terms. Due to this fact, the election between LOWA and ULOWA for aggregating labels in balanced sets should not be tough. It depends on whether one wants to work at a symbolic or a semantic level.

In addition, the ULOWA aggregator is based on the OWA operator and permits to customize the results using different well-known decision-makers policies. The influence of the weight in the final result has also been analysed, and the results obtained by using LOWA and ULOWA have been compared.

3.4 Induced ULOWA operator

This subsection presents the *Induced Unbalanced Linguistic Ordered Weighted Averaging* (IULOWA) operator defined in (Marin et al. 2011a). It is also demonstrated how it fulfils the monotonicity, commutativity, idempotency and boundary conditions usually required in aggregation operators. Like the other OWA operators, IULOWA provides a family of aggregation operators that is parameterized between the linguistic minimum and maximum and that includes a wide range of particular cases

such as the unbalanced linguistic average (ULA), the unbalanced linguistic OWA (ULOWA), the unbalanced linguistic weighted average (ULWA) and many others.

Another important contribution related to this operator is the proposal to use some of the information related to the definition of the labels as an order-inducing criterion in the IULOWA operator. Although the order of the arguments can be decided by taking into account the domain requirements, it is sometimes desirable to take into consideration the amount of information contained in the terms themselves. A method for calculating the set of weights of the arguments by taking into account the degree of uncertainty of the labels is proposed. This allows ordering the arguments by giving priority to more specific values because these represent more precise information. The method uses two well-known measures of fuzzy sets, namely *fuzziness* and *specificity*.

The behaviour of the precision-based IULOWA operator in a case study of a real domain application is demonstrated. Specifically, the results obtained from evaluating the environmental impact produced when sewage sludge coming from wastewater treatment plants is used as fertilizer on agricultural soils have been analysed. In this application, a two-stage aggregation is needed because there is a set of experts that evaluate a set of options using the same set of criteria (i.e. variables). For this reason, the IULOWA operator is further extended by using multi-person techniques in the analysis (Merigó, Casanovas 2011a) and in doing so, the multi-person – IULOWA (MP-IULOWA) operator has been defined. By including the opinions of several experts, more reliable results have been obtained because the decision can be based on the knowledge of a group of people rather than on the opinion of a single individual. Moreover, the use of unbalanced and induced information allows dealing with complex environments where some of the information is more representative and that fact needs to be taken into account in order to correctly assess the aggregation.

3.4.1 Definition of the IULOWA operator

The induced unbalanced LOWA (IULOWA) is an aggregation operator for linguistic values that are defined on an unbalanced vocabulary S . As it is based on IOWA, the operator is able to manage complex decision problems by using order-inducing variables.

Definition 3.6. The Induced Unbalanced Linguistic Ordered Weighted Average, based on the ordering criterion u , is calculated as:

$$\begin{aligned} IULOWA_w(\langle u_1, a_1 \rangle, \langle u_2, a_2 \rangle, \dots, \langle u_m, a_m \rangle) &= \\ &= W \cdot B^T = C^m \{w_k, b_k, k = 1, \dots, m\} = \\ &= w_1 \otimes b_1 \oplus (1 - w_1) \otimes C^{m-1} \{\beta_h, b_h, h = 2, \dots, m\} \end{aligned} \quad (3.9)$$

where B is the induced ordered vector, i.e., $B = (a'_{\sigma(1)}, a'_{\sigma(2)}, \dots, a'_{\sigma(j)})$ where $a'_{\sigma(j)}$ corresponds to the value a_j having the j -th largest u_i . $W = (w_1, \dots, w_m)$ is the usual weighting vector that defines the aggregation

policy of the OWA operator, with $w_i \in [0,1], \sum w_i = 1$. The final convex combination of two linguistic terms is the same as in the ULOWA operator defined in Eq.(3.6).

3.4.2 Properties of the IULOWA operator

As stated in (Yager 1988), an aggregation operator should have the following properties: *monotonicity*, *commutativity*, *idempotency*, and *be a bounded operator* (i.e. “orand”).

Property 1: The IULOWA operator is *increasingly monotonous* with respect to the argument values if the associated order-inducing values remain unchanged:

Let us consider two order-induced vectors $P = \{\langle u_1, p_1 \rangle, \dots, \langle u_m, p_m \rangle\}$ and $Q = \{\langle u'_1, q_1 \rangle, \dots, \langle u'_m, q_m \rangle\}$. If $\forall j, u_j = u'_j$ and $\forall j, p_j \geq q_j$, then $IULOWA_w(P) \geq IULOWA_w(Q)$.

That means, as will be detailed further in this section, that if each term is replaced with another that has the same specificity and fuzziness but a greater preference in the scale S , the result will also be an equal or better term in the preference scale. In fact, the proof of this property may be reduced to the one of the ULOWA operator (Isern et al. 2010), because the inducing variable does not change the order.

Proof: Let $IULOWA_w(P) = IULOWA_w(\langle u_1, p_1 \rangle, \dots, \langle u_m, p_m \rangle)$, and $IULOWA_w(Q) = IULOWA_w(\langle u'_1, q_1 \rangle, \dots, \langle u'_m, q_m \rangle)$. If $\forall j, u_j = u'_j$ and $\forall j, p_j \geq q_j$, any induced permutation of the elements satisfies the condition $\forall j, p'_{\sigma(j)} \geq q'_{\sigma(j)}$, and $IULOWA_w(p'_{\sigma(1)}, \dots, p'_{\sigma(m)}) \geq IULOWA_w(q'_{\sigma(1)}, \dots, q'_{\sigma(m)})$, due to the monotonicity of the ULOWA operator. Then $IULOWA_w(P) \geq IULOWA_w(Q)$. \square

Property 2: The IULOWA operator is *commutative*:

$$IULOWA_w(\langle u_1, a_1 \rangle, \dots, \langle u_m, a_m \rangle) = IULOWA_w(\langle u'_1, a'_1 \rangle, \dots, \langle u'_m, a'_m \rangle),$$

where $(\langle u'_1, a'_1 \rangle, \dots, \langle u'_m, a'_m \rangle)$ is any permutation of the elements in $(\langle u_1, a_1 \rangle, \dots, \langle u_m, a_m \rangle)$.

Proof: The IULOWA operator reorders the arguments according to the order-inducing variable. Thus, if $A = (\langle u_1, a_1 \rangle, \dots, \langle u_m, a_m \rangle)$ is any permutation of $A' = (\langle u'_1, a'_1 \rangle, \dots, \langle u'_m, a'_m \rangle)$, the order induced for A and A' will be the same. Therefore, $IULOWA_w(A) = IULOWA_w(A')$. \square

Property 3: The IULOWA operator is *idempotent* in the sense that $IULOWA_w(\langle u_1, a_1 \rangle, \dots, \langle u_m, a_m \rangle) = a$, if $\forall j, a_j = a$.

Proof: The proof does not depend on the inducing variable, because in this case the values to be aggregated are the same for all the arguments. Then, according to the definition of the IULOWA operator, we have a final step Eq.(3.6) that consists of $IULOWA_w(a_{m-1}, a_m) = s_k$, where $k = w_1 \otimes s_j \oplus (1 - w_1) \otimes s_i = \operatorname{argmax}_{i \leq p \leq j} \{Sim(s_p, \delta)\}$. In this case, $i = j$, so $s_k = s_i = s_j = a$. Recursively, we obtain $IULOWA_w(a_1, \dots, a_m) = a$. \square

Property 4: The IULOWA operator is *bounded*. That is, for any weighting vector W :

$$\min(a_1, \dots, a_m) \leq IULOWA_w(\langle u_1, a_1 \rangle, \dots, \langle u_m, a_m \rangle) \leq \max(a_1, \dots, a_m).$$

Proof: Given $s_j, s_i \in S (j \geq i)$, the convex combination of these two terms has been defined as $C^2\{w_i, b_i, i = 1, 2\} = w_1 \otimes s_j \oplus (1 - w_1) \otimes s_i = s_k$. According to Eq.(3.6), $k = \operatorname{argmax}_{i \leq p \leq j} \{Sim(s_p, \delta)\}$. That is, the resulting label from the combination of two labels is $C^2(s_i, s_j) = s_k$ with $i \leq k \leq j$. This means that a result out of the limits given by the labels that are aggregated at each step cannot be obtained. \square

3.4.3 Families of IULOWA operators

The IULOWA operator permits the definition of a wide range of families of unbalanced linguistic aggregation operators following the methodology used in the OWA literature (Merigó, Casanovas 2011c; Xu 2006). Note that each specific case is useful in certain situations depending on the objectives of the analysis. For example, when aggregating m labels, the following cases can be considered:

- If $w_j = 1/m$, for all j , the unbalanced linguistic average (ULA) is obtained.
- The induced unbalanced linguistic maximum is obtained if $w_1 = 1$ and $w_j = 0$, for all $j \neq 1$, which gives the value a_i , with maximum u_i , because $u_1 = \max\{u_i\}$ after the reordering stage.
- The induced unbalanced linguistic minimum is obtained if $w_m = 1$ and $w_j = 0$, for all $j \neq m$, which gives the value a_i , with minimum u_i , because $u_m = \min\{u_i\}$ after the reordering stage.
- The unbalanced linguistic weighted average (ULWA) is formed if $u_i > u_{i+1}$, for all i .
- The unbalanced LOWA operator is obtained if the j th largest label, s_j , according to the scale S is also ordered at position j according to the inducing variable U , for all j .
- Step-IULOWA: This occurs if there is a position $1 \leq k \leq m$ so that $w_k = 1$ and $w_j = 0$, for all $j \neq k$.
- Median-IULOWA: If m is odd, $w_p = 1$ is assigned, and $w_j = 0$ for all others, with p the position of the $[(m + 1)/2]$ -th largest

u_i . If m is even, for example, $w_p = w_q = 0.5$ and $w_j = 0$ for all others are assigned, with p and q being the positions of the $(m/2)$ -th and $[(m/2) + 1]$ -th largest u_i .

- Olympic-IULOWA: This occurs if $w_p = w_q = 0$, with $u_p \rightarrow \max\{u_i\}$ and $u_q \rightarrow \min\{u_i\}$, and for all others $w_j = 1/(m - 2)$.
- Window-IULOWA: This occurs if $w_j = 1/d$ for $k \leq j \leq k+d-1$ and $w_j = 0$ for $j > k+d$ and $j < k$. Note that k and d must be positive integers so that $k+d-1 \leq m$.
- Centred-IULOWA: This occurs if the aggregation is symmetric, strongly decaying and inclusive. It is symmetric if $w_j = w_{m-j+1}$. It is strongly decaying if when $i < j \leq (m+1)/2$ then $w_i < w_j$ and when $i > j \geq (m+1)/2$ then $w_i < w_j$. It is inclusive if $w_j > 0$ for all j .
- Slide-IULOWA: Three types of this operator can be defined on the basis of its degree of andness (α) and orness (β), where $\alpha, \beta \in [0,1]$ and $\alpha + \beta \leq 1$:
 - Generalised Slide-IULOWA: When $w_1 = (1 - (\alpha + \beta))/m + \beta$, $w_m = (1 - (\alpha + \beta))/m + \alpha$, and $w_j = (1 - (\alpha + \beta))/m$.
 - Orlike Slide-IULOWA: If $\alpha = 0$.
 - Andlike Slide-IULOWA: If $\beta = 0$.

3.4.4 Order Inducing Variables

In this subsection the feature of the IULOWA operator that distinguishes it from the ULOWA operator is analysed; that is, the order-inducing variable that is used in the reordering process of the linguistic labels. With this type of operator, it is possible to deal with complex reordering processes in which the highest linguistic value in S is not the optimal value for the decision maker.

It is divided in two parts: the first part proposes a procedure to obtain the order inducing variable that sorts out the aggregating values, and the second part introduces the way of generating the weights taking into account the uncertainty of the values that are aggregated.

Order induction

As pointed out previously, the order inducing variable can be obtained using different procedures. On one hand, the decision maker can express his/her personal ordering directly on the values of the domain of reference. On the other hand, it is also interesting to have automatic processes for generating the order-inducing criterion. In this latter case, the order is linked to certain features of the set of arguments, such as the distance among the values, the past history of values or the confidence in the values.

In this part a new way of inducing the order that is related to the additional information given by the shape of unbalanced terms is proposed. As was mentioned previously, unbalanced terms permit the definition of linguistic variables with different granularity and distribution for the positive and the negative values.

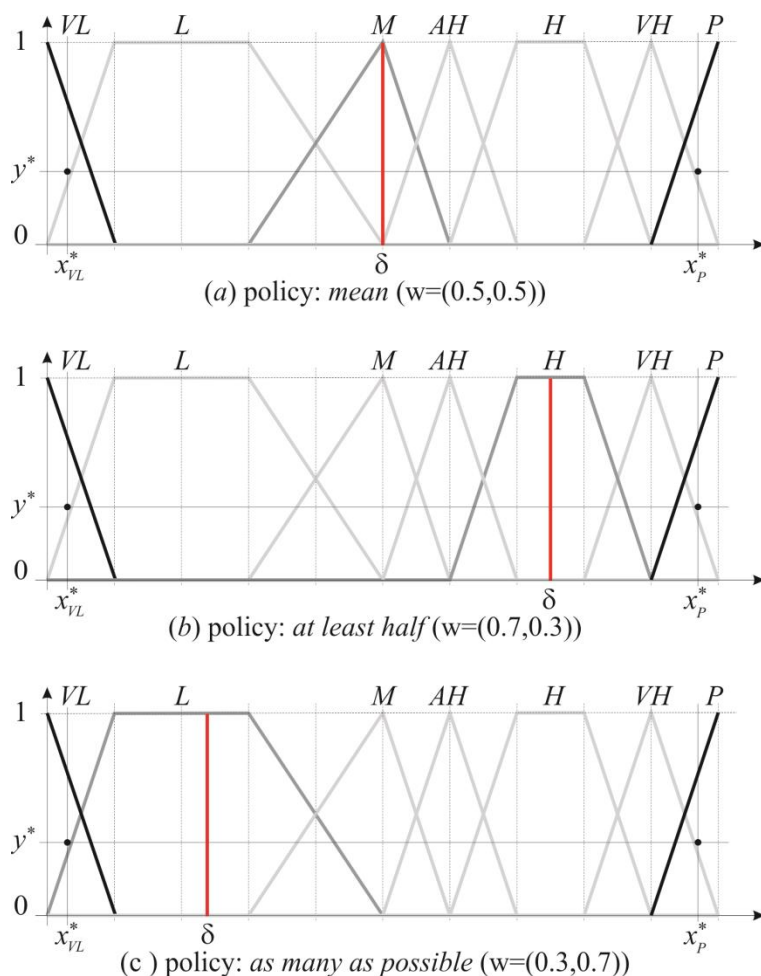


Figure 14. Examples of ULOWA aggregation of two labels (VL and P) when changing the weights

Let us assume that the terms shown in Figure 14 are going to be used to evaluate the performance of a certain object. People usually do not assign extreme values unless they are really sure about the performance of the object; thus, two very precise fuzzy sets for “Very Low” (VL) and “Perfect” (P) (the most negative and most positive terms) have been defined. Being interested in finding objects with good performance, three terms are used to indicate different degrees of positivity (AH , H , VH), while only one to indicate low performance (L). Therefore, L is much more uncertain than the others. Similarly, one can consider that the labels AH (“Almost High”) and VH (“Very High”) are qualifying the term H (“High”), indicating “a little less than High” or “a bit more than High”, respectively. In that way, they are more precise values than High. These specific semantics of the different labels can only be captured using an unbalanced set of terms.

This difference on the certainty of the terms should be taken into account during the aggregation process, as each label is providing a different amount of information about the evaluated alternative. In fact, if it is considered that both triangular and trapezoidal fuzzy sets can be associated to the labels (as in Figure 14), then the uncertainty of the labels is not only related to their support intervals in the reference domain but also to their kernel (*i.e.* the set of points with value 1).

Taking into account the different features of the definition of the linguistic variables pointed out before, the new proposal is to use a measure of the uncertainty of the linguistic labels as the order-inducing criterion for the aggregation. Thus, the arguments will be ordered by decreasing uncertainty. In this way, the contribution of precise labels is prioritized while the effect of uncertain labels is reduced.

In the literature (Bonissone, Decker 1985; Garmendia et al. 2006; Klir 1993; Yager 1990), two types of uncertainty in fuzzy sets are recognized: (1) *specificity*, related to the measurement of imprecision, which is based on the cardinality of the set, and (2) *fuzziness*, or entropy, which measures the vagueness of the set as a result of having imprecise boundaries.

With regards to the measure of *Specificity* (Yager 2008), let X be a set and let $[0,1]^X$ be the class of fuzzy sets on X . A measure of specificity is a function $Sp: [0,1]^X \rightarrow [0,1]$ so that:

1. $Sp(\emptyset) = 0$
2. $Sp(\mu) = 1$ if and only if μ is a singleton
3. If μ and γ are normal fuzzy sets in X and $\mu \subset \gamma$, then $Sp(\mu) \geq Sp(\gamma)$

The following specificity measure, for a fuzzy set A defined on X , is defined as a generalization of other previous formulations (Yager 2008):

$$Sp(A) = T \left(\alpha_{sup}, N \left(\int_0^{\alpha_{sup}} M(A_\alpha) d\alpha \right) \right) \quad (3.10)$$

In this expression T is a T-norm, $\int_0^{\alpha_{sup}}$ is a Choquet integral, α_{sup} the superior α -cut, N a negation operator and M a fuzzy measure.

A special case of Eq.(3.10) is given in Eq.(3.11), by considering the T-norm \min , the standard negation $N(x) = 1 - x$ and the Lebesgue-Stieltjes fuzzy measure $M([a, b]) = b - a$. Taking these parameters and a normalized fuzzy set (with $\alpha_{sup}=1$), the specificity of a fuzzy set defined in the $[a, b]$ interval can be calculated as:

$$Sp(A) = 1 - \frac{\text{area under } A}{b - a} \quad (3.11)$$

With regards to the measure of *fuzziness* (De Luca, Termini 1972), let X be a set and let $[0,1]^X$ be the class of fuzzy sets on X . A measure of fuzziness is a function $Fz: [0,1]^X \rightarrow [0,1]$ so that:

1. $Fz(A) = 0$ if A is a crisp set
2. $Fz(A) = 1$ if $\forall x \in X, A(x) = 1/2$
3. $Fz(A) \leq Fz(B)$ if A is less fuzzy than B , i.e. $A(x) \leq B(x) \leq 1/2$ or $A(x) \geq B(x) \geq 1/2$ for every $x \in X$

The most common way to calculate the fuzziness is in terms of the lack of distinction between the fuzzy set A and its complement A^c . A general definition of this type of fuzziness measure is based on an aggregation operator h and a distance function d , so that:

$$Fz(A) = h_{x \in A} \left(d(A(X), A^C(x)) \right) \quad (3.12)$$

For the case of continuous domains, considering the standard negation operation and the Hamming distance, Eq. (3.12) corresponds to:

$$Fz(A) = 1 - \frac{1}{b-a} \int_a^b |A(x) - 1| \quad (3.13)$$

Specificity and fuzziness refer to two different characteristics of fuzzy sets. Specificity (or its counterpart, non-specificity (Klir, Yuan 1995)) measures the degree of truth of the sentence “containing just one element”. Fuzziness measures the difference from a crisp set. For decision making purposes, it seems desirable to have labels that correspond to single elements, rather than to large sets of values, which may hamper the selection of the appropriate alternative. For this reason, this Thesis proposes to use a measure of specificity as the order-inducing variable in the aggregation of linguistic terms that qualify a set of alternatives in a decision making process.

When there are ties between different terms with the same specificity, a second ordering criterion may be the fuzziness associated to the set. An increasing ordering of fuzziness will be used, as those terms with less uncertainty are preferred. If this second criterion also leads to some ties, a decreasing ordering on the preference scale associated to the terms can be used. Figure 15 shows two fuzzy sets with the same specificity according to Eq.(3.11):

$$Sp(A) = 1 - \frac{\text{area of } A}{b-a} = 1 - \frac{0.2 \cdot 1}{1-0} = 0.9, \text{ and}$$

$$Sp(B) = 1 - \frac{\text{area of } B}{b-a} = 1 - \frac{2\left(\frac{0.05}{2}\right) + 0.05}{1-0} = 0.9,$$

but different fuzziness ($Fz(A) = 0.1$ and $Fz(B) = 0.05$) according to Eq.(3.13), so that:

$$Fz(A) = 1 - \frac{1}{b-a} \int_a^b |A(x) - 1| = 1 - \int_0^1 |A(x) - 1| = 1 - 0.9 = 0.1,$$

$$Fz(B) = 1 - \frac{1}{b-a} \int_a^b |B(x) - 1| = 1 - \int_0^1 |B(x) - 1| = 1 - 0.95 = 0.05.$$

In this example, the set A is fuzzier than B , so B is preferred.

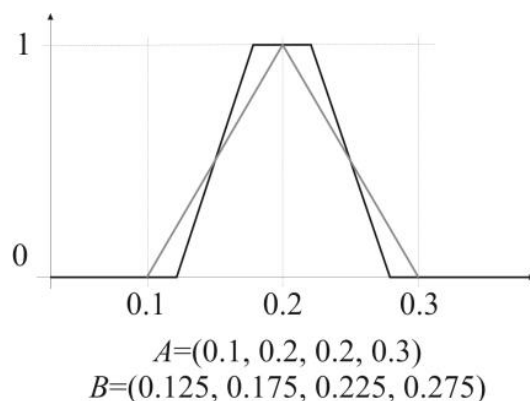


Figure 15. Two fuzzy sets with the same specificity and different fuzziness

Definition 3.7. Precision-based IULOWA: Given a set of unbalanced linguistic arguments $\{a_1, \dots, a_m\}$, the induced aggregation according to the uncertainty of those m terms is calculated by using the IULOWA operator (Eq.(3.9)) where B is the induced ordering vector, so that $B = (b_1, b_2, \dots, b_m)$ satisfies these conditions:

- $\forall k \ 1 \leq k < m, \ Sp(b_k) \geq Sp(b_{k+1})$
- $\forall k \ 1 \leq k < m, \ \text{if } Sp(b_k) = Sp(b_{k+1}), \ \text{then } Fz(b_k) \leq Fz(b_{k+1})$
- $\forall k \ 1 \leq k < m, \ \text{if } Sp(b_k) = Sp(b_{k+1}) \ \text{and} \ Fz(b_k) = Fz(b_{k+1})$ then $b_k > b_{k+1}$ according to the linguistic scale S .

Notice that if the fuzzy sets associated to the terms correspond to crisp numbers, IULOWA is reduced to the OWA operator.

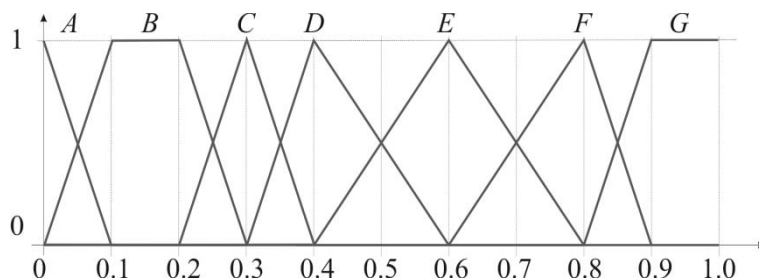


Figure 16. Linguistic variable with 7 terms (test 1)

The following example shows how the terms depicted in Figure 16 would be sorted according to the previous ordering rules. Table 3 shows the information regarding each of the terms needed to conduct the sorting procedure. Specificity is calculated following Eq.(3.11), whereas fuzziness is obtained using Eq.(3.13).

Table 3. Uncertainty measures for the terms in Figure 16

Term	Definition	Specificity	Fuzziness	Index
A	(0.0,0.0,0.0,0.1)	0.95	0.05	0
B	(0.0,0.1,0.2,0.3)	0.80	0.10	1
C	(0.2,0.3,0.3,0.4)	0.90	0.10	2
D	(0.3,0.4,0.4,0.6)	0.85	0.15	3
E	(0.4,0.6,0.6,0.8)	0.80	0.20	4
F	(0.6,0.8,0.8,0.9)	0.85	0.15	5
G	(0.8,0.9,1.0,1.0)	0.85	0.05	6

Taking into account the specificity, the labels are ordered as $A > C > (D, F, G) > (B, E)$. Note that there are two ties: the first one between D, F and G (with $Sp = 0.85$), and the second one between B and E ($Sp = 0.8$). Using the fuzziness measure to solve the ties, G ($Fz = 0.05$) is put before D and F ($Fz = 0.15$) in the first tie, because more priority is given to less fuzzy terms. In the second tie, B ($Fz = 0.10$) goes before E ($Fz = 0.20$). As it can be seen, by measuring fuzziness no decision can be made about the order between D and F, so the index of the terms is used to decide their relative position, putting F (index=5) before D (index=3). Thus, the induced order according to the procedure proposed here is $A > C > G > F > D > B > E$.

Weight generation

As has been said previously, the OWA weights w_i are used to define different conjunction/disjunction aggregation models (Yager 1998; Yager 2009). As proposed in the literature (Cabrerizo et al. 2010; Torra, Narukawa 2007; Chiclana et al. 2007), the inclusion of an additional variable in the OWA aggregator may also involve the transformation of the set of weights.

In this section it is proposed to modify the set of weights associated with the arguments by taking into consideration the uncertainty of the values that are aggregated. The rationale is that the more specific values should have a higher weight, whereas the less specific terms (which may be taken as less reliable) should have a lower weight.

Using the family of fuzzy quantifiers proposed by Yager (Yager 1988), the set of weights associated to a set of terms $\langle u_1, \dots, u_m \rangle$ to be aggregated is obtained with the expression:

$$w_k = Q\left(\frac{S(k)}{S(m)}\right) - Q\left(\frac{S(k-1)}{S(m)}\right), \quad (3.14)$$

where $S(k) = \sum_{l=1}^k u_{\sigma(l)}$ and σ is the permutation according to the order-inducing procedure established before. $Q(p)$ indicates the degree of compatibility of p with the concept denoted by Q . For example, if Q represents a linguistic quantifier such as “most of” and $Q(0.95) = 1$, then it can be said that a value of 95% is completely compatible with the idea conveyed by the linguistic quantifier “most of”.

The properties of the quantifier function must be taken into account in order to generate a coherent set of weights for the OWA operator. Taking the usual quantifier $Q(r) = r^a$ (Yager 1988), if $a \in [0,1]$ then the weighting function is concave, which ensures that the larger the specificity, the higher the weight w_k of the corresponding argument (Chiclana et al. 2007). It is worth noting that with $a \in [0,1]$ the aggregation policy is disjunctive, which means that uncertain evaluations can be replaced with the most specific (and least fuzzy) available values.

Table 4 shows an example of weights obtained without taking into account the specificities. Tests have been done considering several values of the parameter a , ranging from 0.1 (where the result is mostly based on

the first argument) to 1 (which corresponds to an arithmetic average of the arguments, as the weights are equal for all the values).

Table 4. Weights obtained without specificity

a	Weights
0.1	(0.851, 0.061, 0.038, 0.028, 0.022)
0.25	(0.668, 0.127, 0.085, 0.066, 0.054)
0.5	(0.447, 0.185, 0.142, 0.120, 0.106)
0.75	(0.299, 0.204, 0.179, 0.164, 0.154)
1	(0.200, 0.200, 0.200, 0.200, 0.200)

To evaluate the impact of the specificity measure in the set of weights, two tests have been done. The first is based on the linguistic variable with 7 terms represented in Figure 16. The generation of weights is considered for the values (A, C, F, B, B) with specificities (0.95, 0.9, 0.85, 0.8, 0.8) respectively (see Table 3). The results are shown in Table 5.

Table 5. Weights obtained in Test 1

a	Weights
0.1	(0.860, 0.059, 0.036, 0.025, 0.020)
0.25	(0.686, 0.124, 0.080, 0.060, 0.050)
0.5	(0.470, 0.186, 0.136, 0.110, 0.098)
0.75	(0.322, 0.209, 0.174, 0.152, 0.143)
1	(0.221, 0.209, 0.198, 0.186, 0.186)

In this test, the specificities of the terms that are aggregated are very similar. For this reason, the weights in Table 5 are quite similar to those in Table 4 where specificity was not considered. This shows that when the specificity (i.e. confidence) of the terms is similar, the weights are not heavily modified.

For the second test, another set of terms with different degrees of specificity has been used, as shown in Figure 17. In this case the values to aggregate are (E, B, B, C, C), with specificities (0.95, 0.8, 0.8, 0.5, 0.5) respectively.

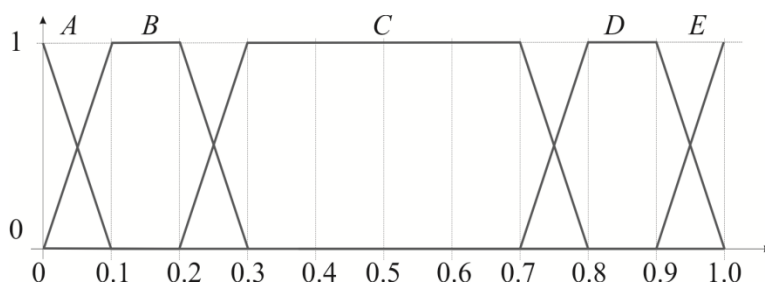


Figure 17. Linguistic variable with 5 terms (test 2)

In this second test the two last terms have a specificity (0.5) much lower than the first three terms (0.95 and 0.8). The results given in Table 6 show that this difference affects the weights as expected, giving more weight to the less uncertain terms. A notable increase in the overall weight of the first three terms and a decrease in the weight of the last two terms can be seen.

Table 6. Weights obtained in Test 2

a	Weights
0.1	(0.876, 0.056, 0.036, 0.017, 0.015)
0.25	(0.719, 0.119, 0.083, 0.042, 0.037)
0.5	(0.517, 0.187, 0.145, 0.079, 0.072)
0.75	(0.372, 0.216, 0.192, 0.112, 0.108)
1	(0.268, 0.225, 0.225, 0.141, 0.141)

3.4.5 IULOWA multi-person multi-criteria case study

In this subsection a real environmental evaluation problem is addressed. In particular, the impact of disposing sewage sludge in agricultural soils is studied. *Environmental Impact Assessment* is defined by the International Association for Impact Assessment (IAIA) as “the process of identifying, predicting, evaluating and mitigating the biophysical, social, and other relevant effects of development proposals prior to major decisions being taken and commitments made”. During the last few decades, the increase of sewage sludge production as a residue of Waste Water Treatment Plants (WWTP) has become an environmental problem in several countries. To maintain sustainability, countries are encouraged to promote the value of sewage sludge as a useful by-product. One of the most widespread practices has been to apply sewage sludge to agricultural soils as fertilizer. Although this option is generally accepted because it reduces fertilizer costs, it may have ecological and human impacts. In the SOSTAQUA Spanish research project, these impacts have been studied and evaluated using many different criteria. Criteria were structured along three basic axes: economic aspects, environmental suitability and human health risks (Valls et al. 2010; Pijuan et al. 2010; Kaya, Kahraman 2011). For sludge managers, the decision on how to distribute the available sludge (from different WWTPs) among their clients (farmers with different agricultural fields) is quite complex due, on one hand, to the large amount of information that has to be considered and, on the other hand, to the expert knowledge that is required to make a correct evaluation. For this reason, it is important to have tools that evaluate the degree of suitability of using a given sewage sludge on different types of soils in order to find the best possible combination.

The focus is put on the problem of obtaining an overall suitability index that evaluates the environmental impact of certain types of sludge on soil. This overall suitability (or impact) is obtained by aggregating the five criteria presented in Table 7. The evaluation of these criteria is not straightforward and different methodologies have been proposed (Valls et al. 2010; Passuello et al. 2011). Moreover, some of the information considered in the evaluation model is subjectively defined by a domain expert, so it is possible to have different opinions from different people.

Table 7. Environmental criteria

Criterion name	Description	Information used
Biodiversity Suitability	Biodiversity is an indicator of the health of ecosystems. Biodiversity can be adversely affected by metal and organic compound contamination depending on the characteristics of the soil.	Metal concentration in the sludge Organic compounds in the sludge Sludge treatment type Organic matter in the soil Soil texture Soil carbonate level
Nitrates Suitability	Contamination of the soil by nutrients should be minimized. Applying sludge containing nitrates to a soil may affect its recommended level of nitrates.	Organic matter in the sludge Sludge treatment type Nitrates available in the sludge Soil texture Nitrates available in the soil
Organic Matter Suitability	Soil organic matter regulates several processes in the soil (e.g.as organic matter mineralizes slowly, nutrients are released at a slower pace, reducing the potential risk of nitrogen leaching to groundwater).	Organic matter in the sludge Organic matter in the soil Sludge treatment type
pH Suitability	Metal contamination in soils is related to its pH. For this reason, basic soils are preferred for sewage sludge treatment. Acid soils should receive sludge with a high pH.	Sludge pH Soil pH
Soil Contamination Suitability	Soil contamination refers to the presence of heavy metals and organic compounds in a soil. The presence of contaminants in sewage sludge may result in risks to humans and ecosystems. The contaminant's movement between environmental compartments may lead to soil contamination.	Metals concentration in the sludge Organic compounds in the sludge Sludge treatment type Organic matter in the soil Soil texture Soil carbonates level Soil pH

The multi-person multi-criteria aggregation process

It is quite common to find problematic decisions in which a set of alternatives are evaluated by different experts on a set of criteria. In this scenario, a two stage process of aggregation is carried out. First, the experts' evaluations of each criterion are fused in order to find a collective result for each criterion. Afterwards, collective criteria are aggregated in order to find the overall evaluation for each alternative. This 2-stage process is illustrated in the following figure.

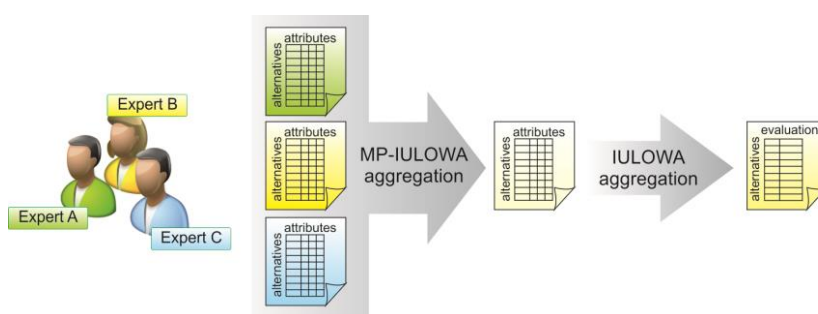


Figure 18. Diagram of the multi-person multi-criteria aggregation process

Definition 3.8. Let $O = \{O_1, O_2, \dots, O_n\}$ be a finite set of options (or alternatives) to be considered in the group decision making problem. Let $C = \{C_1, C_2, \dots, C_m\}$ be a set of criteria (or attributes), forming the payoff matrix of linguistic terms in S , $(a_{ij})_{n \times m}$. Let $E = \{E_1, E_2, \dots, E_q\}$ be a finite set of experts (or decision makers or stakeholders) who participate in the decision making process, so that each expert E_k provides his/her own payoff matrix $(a_{ij}^k)_{n \times m}$. The process can be described as follows:

Step 1: For each option O_i and each criterion C_j , take the q values of the experts E and calculate the weighting vector W to be used in the IULOWA operator, according to the order-inducing variable U (i.e. the specificity and fuzziness of the labels), following the method proposed previously. Then apply IULOWA to aggregate the q values of the experts E using the weighting vector W following the definition of the Precision-based IULOWA. The result is the collective payoff matrix $(a_{ij})_{n \times m}$.

Step 2: For each option O_i and their collective scores obtained in Step 1, calculate the weighting vector W to be used in the IULOWA operator, according to the order-inducing variable U (i.e. the specificity and fuzziness of the linguistic labels of the i -th row of the matrix). Then, calculate the overall aggregated results with the IULOWA operator using again the Precision-based IULOWA.

Step 3: Adopt decisions according to the results found in the previous steps. Select the alternative that provides the best result. Otherwise, establish an ordering or a ranking of the alternatives from the most- to the least-preferred alternative, to enable the consideration of more than one selection.

This double-aggregation process is applied to a given option O_j described with m criteria by q experts, and can be expressed as a function $MP - IULOWA: S^m \times S^q \rightarrow S$ so that:

$$MP - IULOWA\left(\left(\langle u_1^1, a_1^1 \rangle, \dots, \langle u_m^1, a_m^1 \rangle\right), \dots, \left(\langle u_1^q, a_1^q \rangle, \dots, \langle u_m^q, a_m^q \rangle\right)\right) = IULOWA\left(IULOWA_{j=1, \dots, m}\left(\langle u_j^1, a_j^1 \rangle, \dots, \langle u_j^q, a_j^q \rangle\right)\right) \quad (3.15)$$

Solving the case study

In this subsection the MP-IULOWA operator is applied to an example with 3 types of sludge (S1, S2, S3) and 4 agricultural fields (F1, F2, F3, F4), which leads to a total of 12 different combinations or cases. Let us assume that three experts (E1, E2, E3) have evaluated those cases with the five criteria explained in

Table 7 and using the unbalanced linguistic variable depicted in Figure 19.

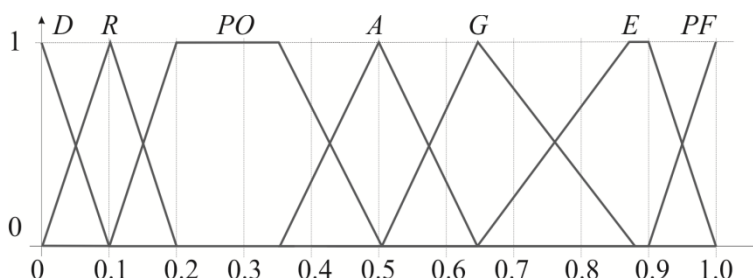


Figure 19. Evaluation scale for the criteria (D: “Dangerous”, R: “Risky”, PO: “Poor”, A: “Acceptable”, G: “Good”, E: “Excellent” and PF: “Perfect”)

The linguistic vocabulary gives 7 degrees of suitability, ranging from a dangerous situation to a perfect suitability evaluation. This linguistic scale presents an unbalanced set of terms with different specificity and fuzziness (see Table 8). The most specific terms are those that correspond to the most extreme scores (“Dangerous” and “Perfect”), followed by the term “Risky”. This specificity is needed because those labels refer to very critical and precise situations. A not so specific neutral term “Acceptable” is available for use if there is a combination of values that is neither positive nor negative, from the point of view of environmental suitability. The other terms permit the identification of different suitability levels without needing to be too precise.

Notice that, in this vocabulary, the specificity of the terms is a useful indicator for inducing the weights of the aggregation process because the most specific values correspond to those terms that are detecting the most interesting situations, from the decision maker’s point of view. In fact, the most specific terms give more information than the rest. It is also necessary to take into account that when the specificity of the evaluations is the same, the fuzziness is used to solve those ties. In this example it is assumed that the environmental experts that evaluate the sludge samples have similar expertise, so it is not necessary to assign different confidences to each of them.

Table 8. Definition and values of specificity and fuzziness of the linguistic terms

Linguistic value	Definition	Specificity	Fuzziness
Dangerous	(0.0, 0.0, 0.0, 0.1)	0.950	0.050
Risky	(0.0, 0.1, 0.1, 0.2)	0.900	0.099
Poor	(0.1, 0.2, 0.35, 0.5)	0.725	0.125
Acceptable	(0.35, 0.5, 0.5, 0.65)	0.850	0.150
Good	(0.5, 0.65, 0.65, 0.875)	0.812	0.187
Excellent	(0.65, 0.875, 0.9, 1.0)	0.812	0.162
Perfect	(0.9, 1.0, 1.0, 1.0)	0.950	0.050

The following tables (Tables 9, 10 and 11) correspond to the three experts’ evaluations of the twelve cases, taking into account the environmental criteria explained above.

Table 9. Evaluation of expert A

Case	Biodiversity	Nutrients suitability	Organic matter suitability	pH suitability	Absence of soil contamination
1	Acceptable	Dangerous	Risky	Acceptable	Risky
2	Good	Acceptable	Perfect	Perfect	Poor
3	Poor	Dangerous	Acceptable	Risky	Risky
4	Acceptable	Acceptable	Good	Perfect	Poor
5	Risky	Acceptable	Acceptable	Excellent	Good
6	Poor	Excellent	Good	Good	Poor
7	Good	Good	Good	Perfect	Acceptable
8	Excellent	Good	Excellent	Excellent	Good
9	Risky	Poor	Acceptable	Dangerous	Acceptable
10	Acceptable	Good	Good	Risky	Poor
11	Good	Good	Excellent	Good	Good
12	Poor	Acceptable	Acceptable	Perfect	Poor

Table 10. Evaluation of expert B

Case	Biodiversity	Nutrients suitability	Organic matter suitability	pH suitability	Absence of soil contamination
1	Risky	Poor	Risky	Excellent	Risky
2	Good	Good	Good	Perfect	Poor
3	Good	Dangerous	Poor	Risky	Risky
4	Risky	Poor	Excellent	Good	Good
5	Acceptable	Excellent	Risky	Good	Acceptable
6	Good	Good	Good	Good	Dangerous
7	Poor	Good	Poor	Perfect	Good
8	Excellent	Acceptable	Excellent	Excellent	Good
9	Dangerous	Good	Acceptable	Dangerous	Acceptable
10	Acceptable	Good	Good	Dangerous	Poor
11	Excellent	Good	Excellent	Good	Good
12	Poor	Acceptable	Acceptable	Perfect	Poor

Table 11. Evaluation of expert C

Case	Biodiversity	Nutrients suitability	Organic matter suitability	pH suitability	Absence of soil contamination
1	Excellent	Poor	Risky	Good	Dangerous
2	Good	Acceptable	Perfect	Perfect	Dangerous
3	Acceptable	Dangerous	Poor	Acceptable	Risky
4	Poor	Acceptable	Good	Perfect	Poor
5	Dangerous	Acceptable	Acceptable	Perfect	Perfect
6	Poor	Good	Excellent	Acceptable	Poor
7	Acceptable	Acceptable	Good	Perfect	Excellent
8	Excellent	Good	Excellent	Perfect	Good
9	Risky	Poor	Poor	Dangerous	Acceptable
10	Risky	Perfect	Good	Risky	Poor
11	Acceptable	Good	Good	Good	Acceptable
12	Poor	Poor	Acceptable	Perfect	Poor

After obtaining the evaluations of the three experts, it is necessary to aggregate them into a single matrix to represent the group opinion regarding the alternatives for the five criteria. As more confidence needs to be given to values with high precision, the proposed two-stage IULOWA aggregation process will be used (see Figure 18).

First, when the three experts' evaluations are aggregated to obtain a single evaluation for each attribute of each alternative, equations Eq.(3.11) and Eq.(3.13) are applied to give more confidence to the labels with more specificity and less fuzziness. Table 12 shows the matrix obtained after the aggregation of the three experts' opinions using the IULOWA operator induced by the specificity and with the quantifier $Q(r) = r^{0.5}$, which corresponds to high *orness*. This policy establishes that the evaluations given by the more uncertain values will be almost ignored, and the overall result will be mostly based on the most specific evaluation given by one of

the experts (w_1 around 0.55). For example, for alternative 1 and the “Biodiversity” criterion, the ranking of the values given by the three experts is “Risky” \succcurlyeq “Acceptable” \succcurlyeq “Excellent”. The result given by IULOWA is Poor, and is mainly based on the combination of the two first labels ($w_1 + w_2 \approx 0.85$). Thus, as the most precise expert has indicated only a “Risky” level of suitability, and taking into account the precise medium evaluation given by “Acceptable”, the result is “Poor”.

In the resulting matrix, which contains the collective evaluation, IULOWA is applied again to each row in order to obtain a final overall evaluation for each alternative. The same aggregation policy is followed, which weights the contribution of the values according to their precision. The last column of Table 12 shows the overall suitability of each of the twelve alternatives considered.

Table 12. Collective data matrix, including the overall suitability value

Case	Biodiversity	Nutrients suitability	Organic matter suitability	pH suitability	Absence of soil contamination	Overall suitability
1	Poor	Risky	Risky	Acceptable	Risky	Risky
2	Good	Acceptable	Excellent	Perfect	Risky	Good
3	Acceptable	Dangerous	Acceptable	Risky	Risky	Risky
4	Poor	Acceptable	Excellent	Excellent	Acceptable	Good
5	Risky	Acceptable	Poor	Excellent	Excellent	Acceptable
6	Acceptable	Excellent	Excellent	Acceptable	Risky	Acceptable
7	Acceptable	Acceptable	Acceptable	Perfect	Acceptable	Good
8	Excellent	Acceptable	Excellent	Excellent	Good	Good
9	Risky	Acceptable	Acceptable	Dangerous	Acceptable	Poor
10	Poor	Excellent	Good	Risky	Poor	Acceptable
11	Acceptable	Good	Excellent	Good	Acceptable	Good
12	Poor	Acceptable	Acceptable	Perfect	Poor	Good

As indicated above, evaluating the environmental impact of treating soil with sewage sludge is quite a delicate task and the experts will want to give more importance to cases where they have detected extreme values such as “Dangerous” or “Perfect”. Using the IULOWA weighting mechanism and applying a disjunctive aggregation policy, it can be seen that the most specific label contributes 45% to the final result ($w_1 = 0.45$). The remaining 55% is divided among the other labels, in particular the one in the second position after the ranking according to the uncertainty. For example, in case 8, the ranking is “Acceptable” \succcurlyeq “Excellent” \succcurlyeq “Excellent” \succcurlyeq “Excellent” \succcurlyeq “Good”, so the final result is mainly a combination of “Acceptable” and “Excellent”, which gives the result “Good”.

Notice that, with this criterion, a precise evaluation is considered the most important because the experts need to be more confident when giving a more specific evaluation than when giving a less specific one. This rationale is clearly exemplified in cases 7 and 9, where an extreme evaluation (positive “Perfect” for case 7 and negative “Dangerous” for case 9) has direct repercussions on the final overall suitability evaluation. In case 7, despite having an “Acceptable” evaluation for most of the attributes, the fact of having a single but very specific “Perfect” evaluation makes the overall evaluation “Good”. A similar event occurs in case 9, where having a “Dangerous” evaluation makes the final evaluation decrease to “Poor”, although most of the attributes have an “Acceptable” suitability.

3.4.6 IULOWA Conclusions

This subsection has presented a new aggregation operator called IULOWA, which enables complex reordering processes to be carried out by using order-inducing variables. In particular, the IOWA operator has been extended to deal with linguistic variables that use unbalanced fuzzy sets. Unbalanced sets of terms allow different degrees of uncertainty to be dealt with in the values that are aggregated, thus permitting sets of linguistic terms to be designed for variables that, due to their nature, require different degrees of precision in different parts of the domain.

A new procedure for aggregating terms with different degrees of precision has been proposed. This process is based on the extension principle and uses operations on the fuzzy sets associated to the linguistic terms that are aggregated. The procedure is recursive, following the well-known LOWA operator. Then, the use of induced variables in unbalanced sets of linguistic terms has been carefully analysed. A procedure to use the measurement of uncertainty as an order-inducing criterion in the IULOWA has been proposed. This approach means that the decision is based on the less uncertain values, which in turn give the decision maker more confidence. The concept of minimum uncertainty is interpreted as maximum specificity and minimum fuzziness, two well-known measures in fuzzy theory. Ties are solved by taking the linguistic scale of evaluation as the preference degree. It has also been shown that it is useful to modify the weighting policy according to the level of uncertainty to make a coherent aggregation of the values.

It can be clearly seen that this new operator includes the ULOWA operator when all the terms have the same specificity and fuzziness. It can also be reduced to the LOWA operator if the terms are balanced. In fact, the IULOWA operator provides a wide range of families of unbalanced linguistic aggregation operators following the methodology used in the OWA literature.

On the basis of the IULOWA operator, a multi-person multi-criteria scenario has been presented, proposing a solution to the decision making problem in two steps: 1) using the IULOWA to obtain a collective value for each criterion of each alternative; and 2) using the IULOWA to combine the aggregated values of the different criteria into a single overall evaluation. The exploitation of the final overall linguistic value will lead to the solution of the problem, that is, it will identify the best alternative/s. This model has been used in a real environmental assessment problem, using a set of criteria defined in the Spanish research project SOSTAQUA. The results obtained show that when specific values give more information than the more uncertain ones, the IULOWA operator and the weighting policy proposed give good and consistent results.

3.5 Conclusions

The motivation, definition and evaluation of the two contributions regarding aggregation operators done in this Thesis have been explained in this chapter: the ULOWA and the IULOWA aggregation operators.

The ULOWA has proven to be a useful extension of the LOWA operator to allow dealing with an unbalanced set of terms. This fact enables defining different degrees of uncertainty to be dealt with in the values that are aggregated, thus permitting sets of linguistic terms to be designed for variables that, due to their nature, require different degrees of precision in different parts of the domain. The other operator described in this chapter, the IULOWA, enables complex reordering processes to be carried out by using order-inducing variables. In particular, the IOWA operator has been extended to deal with linguistic variables that use unbalanced fuzzy sets.

Those two tools provide a very adequate way of aggregating the terms that represent the preferences used to evaluate the attribute values. Consequently, they play a very important role in the whole recommendation process facilitating the process of alternatives rating and ranking. The next chapter is focused on the next part of the system: the preference adaption process.

Chapter 4

Preference Learning

A big challenge on recommender systems is the ability to know exactly why a user selects one alternative instead of the rest of them. This chapter presents a novel approach to infer this information and then adapt the user's interests, described in the user profile, over time.

A set of alternatives can be considered in a different way by different people because each one has its own interests. When a set of ranked alternatives is presented to the user, two things can happen: (a) the user selects the first ranked alternative or (b) the user selects any other alternative. The first case means that the recommendation process has worked accurately, since the system gave the first place to the selected alternative. However, in the second case, there were other alternatives (which can be called *over ranked*) that were considered by the system as better than the one the user finally selected. Thus, that is probably indicating that the information in the user profile is not accurate enough and should be modified. In a nut shell, the main intuition behind the user profile change algorithm is that we should increase the preference on the attribute values present in the selected alternative and decrease the preference on the attribute values appearing in the over ranked alternatives.

Important pieces of information can be extracted from the interaction of the user with the system. They are called *relevance feedback*, and are required by the learning algorithms to improve the user profile. There are two kinds of relevance feedback: the explicit feedback and the implicit feedback.

Explicit feedback is obtained when users are required to evaluate items, indicating how relevant or interesting they are to them using some numeric or linguistic scale. These systems offer high performance and simplicity, as shown in (Morales-del-Castillo et al. 2009; Morales-del-Castillo et al. 2010; Noppens et al. 2006). However, explicit feedback has some serious limitations: the user must spend some time and effort, the rating action distracts the attention of the user from his/her standard workflow, and numeric scales may not be adequate to describe the reactions humans have to items (Fan et al. 2005). Moreover, users are usually reluctant to spend time giving explicit feedback and only 15% of the users would supply it even if they were encouraged to do so (Pazzani, Billsus 1997).

Implicit feedback is obtained by monitoring the user actions and automatically inferring the user preferences. The amount of collected data is consequently very large, the computation needed to derive the profile adaptations is extensive, and the confidence in their suitability is likely to be relatively low. This approach has been less explored, although some

existing methods have shown promising results (Baltrunas, Amatriain 2009; Cheng et al. 2006; Eyharabide, Amandi 2012; Montaner et al. 2003).

One of the objectives of this Thesis is to build automatic and dynamic learning techniques that allow preference learning without requesting explicit information from the user, so the model of relevance feedback used in this work is implicit. For each interaction with the user, the relevant data to update the user profile are the over ranked alternatives and the selected one.

Preferences are learned in different ways on numerical and categorical attributes, as described in the following sections of the chapter. First of all, a brief review comments the main current works on this field and compares them to this work. Then, the management of numerical preferences followed by the management of categorical ones are presented. Although these two types of data are separately explained, an approach that permits the management of both types of data together is also introduced. Finally, the chapter ends with a list of conclusions.

4.1 State of the art

Traditionally, recommender systems create static representations of the user profile based on a predefined set of criteria. Machine Learning techniques are often used to build a representation of the users' preferences from an initial set of data, for instance, creating a vector of weights that represents their interests on a set of concepts (*e.g.* (Kelly, Teevan 2003; Batet et al. 2012)) or applying natural language processing tools to textual data in order to correlate the words representing documents and the preferences of users (*e.g.* (Pham et al. 2012)). These systems have a high confidence in their recommendations because they are based on domain-dependent information, but they lack flexibility to be extended to other domains (Adomavicius, YoungOk 2007), prompting the need for a domain-independent mechanism that can automatically learn the user's preferences.

It is not new to use implicit information, obtained by monitoring the actions of the user, in the adaptation of the knowledge about his/her preferences. There are many different kinds of actions that can be monitored, and the actions may be applied on different types of objects. A comprehensive survey of implicit feedback techniques along these two axes may be found in (Kelly, Teevan 2003).

The next subsections are focused on four key aspects of the recommender systems that try to learn dynamically the preferences of the user from the analysis of implicit information: the number of criteria used in the representation of the domain items, the use of domain-dependent information, the possibility of modelling preferences that change over time, and the use of accumulated historical information. Finally, a discussion on the general shortcomings of current implicit adaptation methods, which are overcome by the algorithms reported in this work, is included.

4.1.1 Number of criteria

It has been pointed out by several researchers (Adomavicius, YoungOk 2007; Weakliam et al. 2005) that most of the recommender systems that include dynamic adaptation techniques defined up to now consider only the case in which each of the options or alternatives to be analysed by the system is defined in terms of one single attribute or criterion (*e.g.* recommending a hotel taking into account only the price per night).

Multi-criteria decision making tools are appropriate when dealing with different criteria at the same time (Valls et al. 2009; Moreno et al. 2013). The main issue is to find a function that permits to combine the user's preferences on all attribute values to determine the overall preference on a given item. One work that considers the multi-criteria case is presented in (Adomavicius, YoungOk 2007). The authors propose two user profile adaptation algorithms. One of them employs collaborative filtering techniques, which fall outside the scope of this work (as we are considering only the problem of learning the preferences of a single user from the analysis of his/her individual interaction with the system). The other one suggests the following steps: to learn the rating prediction for each criterion individually, to learn an aggregation function that puts together the evaluation of each criterion to have a global assessment of an alternative, and finally to use the results of the previous steps to predict the overall rating of the option set. In the work presented in this Thesis, each attribute is also considered individually, and the evaluation of each attribute is aggregated to get the overall rate of an alternative. However, there are two basic differences in favour of the proposal presented in this Thesis. First, as commented previously, our proposal provides a great flexibility in the definition of the mechanism that aggregates the individual preferences of the attributes, permitting its use in very different settings. Moreover, unlike that proposal, we provide a complete and detailed algorithm for the adaptation of the user profile after the analysis of each selection of the user, which specifies precisely how the preferential information on the user profile changes depending on the object selected by the user after each recommendation step.

Another proposal that considers several criteria is made by (Arias et al. 2011). In this case the aim is to filter the news that can be interesting for the user, taking into account the criteria of aboutness, coverage, novelty, reliability and timeliness. The user profile contains information on the preferences of the user in these aspects, and they are modified with the analysis of the news marked by the user as relevant. The authors define a very specific adaptation function for each of the considered criteria, whereas in our approach all the criteria are modified with the same process. This is the main difference of our work with respect to most of the previous works on preference learning, which are domain-dependent and rely heavily on the particular specificities of the domain in order to define the learning algorithm, making them hardly reusable in any other domain. Our proposal is fully domain independent and could be easily applied to any domain defined on numerical and categorical attributes, without having the need to study the concrete attributes of the domain and to define specific preference learning mechanisms adapted to their particularities. Our most restrictive requirement is that the user has to face the selection problem very often, so that his/her continuous selections provide the adaptation

algorithms with the feedback they need to learn quickly and efficiently the user's preferences. However, as argued in the initial section of this Thesis, nowadays users are constantly confronted with a high quantity of selection options and have to take decisions with dozens or hundreds of alternatives continuously, so it can be argued that the adaptation algorithms presented in this work could indeed be useful in many daily decisional problems.

4.1.2 Use of domain-dependent information

The recommender systems that employ implicit adaptation techniques rely on an analysis of the actions made by the user. Each of those actions provides a specific piece of information, known as relevance feedback, which can be leveraged to improve the knowledge about the preferences of the user. Unlike the approach presented in this Thesis, which is generic and domain-independent, most recommender systems focus on a particular domain of application. This decision enables those systems to make a very accurate analysis of the interaction of the user with the system and to make very precise and fine-grained changes in the user profile. However, these systems are hardly reusable and require a strong design and implementation effort, as they imply a very exhaustive previous manual analysis of all the possible user actions in the domain and the *ad-hoc* determination of how each action affects the user profile. Some prominent examples of this kind of systems are provided in the following paragraphs.

For instance, (Joachims, Radlinski 2007) employ the user preferences to sort the results provided by a search engine to a query, using a basic assumption very similar to the one of this work. They have conducted tests that prove that users scan the search results in the order given to them. Thus, they infer that, if a user clicks on a result, that act is providing a double feedback: a positive one, showing that the user is interested in the content of that result, and a negative one, as the system can assume that the user is not interested in the results that were shown above the selected one. Thus, they interpret this feedback in terms of binary preference relationships (result *A* is preferred to result *B*), and they feed a Support Vector Machine with this knowledge to update the user profile.

Another domain-dependent method for profile adaptation (for music recommendation) is presented in (Noppens et al. 2006). It combines explicit ratings given by the users (for each criteria of each item) with implicit information (the time spent hearing each song, and the number of times the user hears each song), and it also incorporates collaborative filtering techniques based on groups of users with similar tastes. In this domain, the recommender presented by (Liu 2012) considers different features of music contents; the novelty is the use of genetic algorithms to store the preferences and then recommend items to users.

Just to give another domain-dependent example, (Weakliam et al. 2005) analyse the actions performed by the user on a Geographical Information System to reflect in the user model which are the particular spatial features in which the user is interested.

As seen on Chapter 2, some systems rely on a domain ontology to guide the recommendation process. Some approaches in that area, such as (Codina, Ceccaroni 2010), also extend the user profile with inference mechanisms that exploit the ontology hierarchy to discover new knowledge regarding the user preferences. In (Borràs et al. 2012) a spreading algorithm to propagate the user preferences through an ontology is presented. Moreover, the authors also introduce an uncertainty factor associated to the interest scores that allows modelling their confidence.

Another recent example of the use of implicit domain-dependent information to adapt the user profile can be found in (Castellano et al. 2010). The authors present a fuzzy framework in which the resources to recommend are tagged with metadata that describe their most prominent features. The user profile describes, using fuzzy sets, the preferences of the user on a set of criteria. A matching mechanism that uses the membership functions associated with the user profile and the resources permits the evaluation of the similarity of both elements and the recommendation of the most similar item. In addition, an adaptation mechanism helps to update the user profile by taking into account the features of the selected resource. However, one of the drawbacks of this framework is the need for tagged information attached to each resource. This implies a subjective annotation of all the resources, taking into account the available criteria.

4.1.3 Dynamic preferences

Recommender systems often assume that the preferences of the user are static and do not change over time. They try to assess these preferences either at the beginning of the recommendation session (*e.g.*, by presenting explicit questionnaires to the user) or during the process of recommendation, trying to learn the user profile from the combination of explicit and implicit information. However, they normally do not consider the case in which the preferences of the user may change dynamically over time. The adaptation algorithm presented in this work allows refining the knowledge about the preferences of the user, and also to change the profile if the user interests changes (as shown in the results presented further in this chapter). This latter aspect has not been considered in many works. One notable exception is presented by (Sigurbjörnsson, Van Zwol 2008), in which a combination of a short-term profile and a long-term profile is suggested. The changes in the profiles are proposed taking into account the last action done by the user (in the short-term one) and a past history of performed actions (for the long-term one). In that work, focused in suggesting Web pages, the user profile is represented through a taxonomy of terms labelled with probabilities, which evolve over time.

4.1.4 Management of user historical data

The system presented in this Thesis stores the selections made by the user over time and analyses them to learn his/her preferences. This management of historical data has similarities with *case based reasoning* (CBR), where cases, that store previous problems and their solutions, are used to help to answer in an efficient way the new problems posed to the system (Lenz 1996). In a nutshell, a CBR system looks for the case that has a stronger similarity with the current problem, retrieves its solution and adapts it to fit

the characteristics of the new situation. As an example of a CBR-based recommender system, *Happy Movie* (Quijano-Sánchez et al. 2011; Quijano-Sánchez et al. 2012) is a Facebook application that provides a group recommendation for a set of people that wish to go together to the cinema. A content-based recommender analyses the features of the movies and compares them with the preferences of the group members. *Happy Movie* uses the group meetings as a case base for further recommendations. Unlike those recommenders based on past cases, it has to be noted that the system proposed in this Thesis does not process previous recommendations and user selections when it has to provide a new recommendation. That historical information has been previously analysed to update the user profile, and this structure is the only one that is used to rank the recommendation alternatives.

A related approach was proposed in the *MyMedia* project, which was one of the first attempts to create a general framework for recommender systems (Marrow et al. 2009). This system was designed as an open infrastructure adoptable in general domains, although it was applied to the recommendation of user-tagged multimedia content. It offers content-based and collaborative-based recommendations algorithms that consider implicit feedback (concretely, the tags given by users to items). *MyMedia* pays special attention to the annotation of the multimedia elements, as this information is used to compare the preferences of the user with the instances to recommend (Gantner et al. 2009; Symeonidis et al. 2010).

The approach in this Thesis is also similar to recent systems (*real-time Web-based recommenders*) that use the Web as a corpus from which information about a set of products can be retrieved and analysed in order to come up with suitable recommendations (Garcia Esparza et al. 2012). For example, *Blippr* is a Twitter-based product recommender that collects dozens of reviews of applications, music, movies, books and games (Phelan et al. 2011a). After an analysis of these reviews, which provide implicit information about the preferences of the users, the system is able to rate those products for an individual or for a community of users. These systems are less susceptible to manipulation and more responsive to searcher needs and preferences. Another example of this type of systems is *Buzzer*, a content-based recommender of news that is capable of analysing the conversations that are taking place on Twitter (Phelan et al. 2009). This system rates RSS news stories by mining trending terms from both the public Twitter timeline and from the timeline of tweets generated by a user's own social graph. The authors of *Buzzer* consider the position of the news selected by the user within the list of recommended items as a measure of the successfulness of the recommender (Phelan et al. 2011b), as it is done further in this document. The main limitation of these systems is the bias imposed by the limited number of terms used in each tweet and a limited accuracy of the recommendations due to the short corpus of past episodes taken into consideration.

4.1.5 Final discussion

In summary, it can be argued from the analysis of the state of the art that most of the approaches to the dynamic implicit adaptation of the user profile have the following shortcomings:

- It can be observed that the definition of techniques applicable in the case in which the alternatives are defined on multiple criteria is scarce. This case is very complex, since the selection of an item by a user does not allow making a direct inference on the preferred value of the user for each of the attributes. However, the results presented further in this document show that it is indeed possible to discover the preferences of the user for each criterion if we consider settings in which the user is constantly faced with decision problems, so the amount of selected (and rejected) alternatives is large and the adaptation system can, after some iterations, tend towards the right profile. We think that these continuous iterative decision problems are the norm, rather than the exception, as exemplified by the multitude of decisions we must take repeatedly on a regular basis (*e.g.*, which news to read every morning, which scientific papers to read every week, which messages to read in our favourite social network every day, which TV channel to watch every evening, etc.).
- Most of the existing techniques for implicit dynamic adaptation of the user profile are heavily centred on a particular domain of application, which implies a detailed analysis of the actions on the domain and the definition of heuristics that permit to infer changes on the user profile from them. Although they can provide very accurate and precise results for a specific problem, this kind of proposals are very difficult to generalize and to apply to other domains. The adaptation algorithms proposed in this paper do not use any kind of domain-dependent knowledge, and they only assume that the user is able to select his/her favourite alternative in each recommendation problem; therefore, they are directly applicable to any domain.
- Finally, most of the systems that include some kind of dynamic learning technique of the user preferences seem to assume that they are static, and do not change over time. The adaptation algorithms proposed in this paper do not make this strong assumption, and can be perfectly used in settings in which the user interests may vary over time.

In summary, the main novel aspects of the dynamic adaptation approach reported in this Thesis, which represent an improvement from most of the existing techniques, are the following: the consideration of multiple criteria (categorical and numerical) to define the recommendation options, the use of generic techniques that can be directly applied in any domain, and the avoidance of the static preferences assumption. As a final remark in this discussion, it can also be noted that most of the recommendation systems that consider numeric criteria translate them to a linguistic domain. This translation eases the management of criteria and permits a generalization of the data but at the same time it adds an error that can be avoided with our approach, in which we manage directly the numerical values.

4.2 Preference learning over numeric preferences

This section presents the contribution designed during the elaboration of this Thesis regarding the dynamic learning of preferences over numerical attributes. At first, it consisted basically in learning the value of maximum preference of the attribute. Then, the procedure was improved to allow learning a more complex preference function, which included, in addition to the value of maximum preference, the two delta values (left and right) and the two slope values (left and right), according to the definition of the numeric preference function in section 2.2.

The numeric adaptation of the user profile is inspired by Coulomb's Law: *"the magnitude of the electrostatics force of interaction between two point charges is directly proportional to the scalar multiplication of the magnitudes of charges and inversely proportional to the square of the distances between them"*. The main idea is to consider the value of maximum preference stored in the profile (current preference) as a charge with the same polarity as the values of the same criterion on the over ranked alternatives, and with opposite polarity to the value of that criterion in the selected alternative. Thus, the value of the profile is "pushed away" by the values in the over ranked alternatives and "pulled back" by the value in the selected alternative.

Two stages have been considered in the adaptation algorithm. The first one, called *on-line* adaptation process, is performed each time the user asks for a recommendation and there are enough over ranked alternatives. The other stage, called *off-line* process, is performed after a certain amount of interactions with the user.

4.2.1 On-line adaptation process

For the on-line stage, the information available in each iteration (or relevance feedback) is the user selection and the set of over ranked alternatives. Moreover, two parameters have been introduced in this process to avoid unexpected behaviours. The first one defines a minimum number of over ranked alternatives (*mo*) needed before proposing any change. The second one (maximum change over profile preferences, *mc*) defines an upper limit on the amount of change that can be applied on the preferred value of a criterion, based on a percentage of its range. With this parameter we avoid abrupt preference changes in a single iteration.

The on-line step calculates the appropriate attraction/repulsion forces for each criterion. This stage takes into account only the selected alternative and the set of over ranked alternatives. However, when the number of over ranked alternatives does not reach a certain value (*mo*), the process does not make any changes in the profile and the current over ranked alternatives are stored for the next execution of the off-line adaptation process (see section 4.2.2). The request of this minimum number is based on the fact that if we have very few over ranked alternatives, the possible changes in the profile are deduced from a small amount of information, making the whole reasoning process unreliable.

In order to calculate the change of the value of preference in the user profile for each criterion it is necessary to study the attraction force done by the selected alternative ($F^s = \{F_1^s, F_2^s, \dots, F_k^s\}$) and the repulsion forces done by the over ranked alternatives ($F^o = \{F_1^o, F_2^o, \dots, F_k^o\}$) in each criterion [1...k]. This is represented in the example in Figure 20, in which the j -th value of the five over ranked alternatives o^0, o^1, o^2, o^3 , and o^4 causes a repulsion force F_j^o , and the value for the same criterion of the selected alternative, s_j , causes an attraction force F_j^s . Both forces are applied on the j -th value of the profile, P_j .

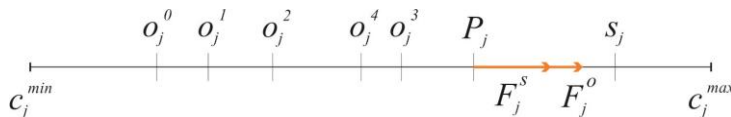


Figure 20. Attraction and repulsion forces, namely F^s and F^o respectively

The attraction force F^s done by the selected alternative for each attribute j is defined as:

$$F^s(P, s, j, \alpha) = \begin{cases} range(c_j) \left(\frac{1}{|s_j - P_j|^\alpha} \cdot \frac{s_j - P_j}{|s_j - P_j|} \right) & \text{if } s_j \neq P_j \\ 0 & \text{if } s_j = P_j \end{cases} \quad (4.1)$$

In this equation, the range of the criterion j ($c_j^{\max} - c_j^{\min}$) is used as a normalization factor of the resulting force, s_j is the value of the criterion j in the selected alternative and P_j is the value of the same criterion in the stored profile P . The parameter α adjusts the strength of the force in order to have a balanced adaptation process. Its influence is more deeply studied further in this section. The goal of this equation is to consider a force when the current profile and the selection differ. For this reason, when those values coincide, the resulting force is 0.

The repulsion force exerted by the over ranked alternatives for each criterion j is defined as a generalization of Eq.(4.1) as follows:

$$F^o(P, \{o^1, \dots, o^{no}\}, j, \alpha) = range(c_j) \sum_{i=1}^{no} \frac{1}{|P_j - o_j^i|^\alpha} \cdot \frac{P_j - o_j^i}{|P_j - o_j^i|} \quad (4.2)$$

In this expression o^i is the i -th over-ranked alternative, o_j^i is the value of attribute j for o^i , and no is the number of over ranked alternatives. In this expression it is assumed that the value of the profile for the criterion is different from the values of that criterion in the over ranked alternatives ($\forall i P_j \neq o_j^i$). If there are alternatives in which the values coincide, those alternatives are omitted from the addition and they do not participate in the determination of this repulsion force.

Finally, both forces are summed up and the resulting force is calculated as:

$$F_j^{\text{on-line}} = F_j^o + (n \cdot \beta) \cdot F_j^s \quad (4.3)$$

The constant $(n \cdot \beta)$ is a correction factor on the attraction force of the selected alternative, where n is the total number of alternatives and β is a parameter that adjusts the strength of the forces in a way similar to the parameter α . It basically regulates the relationship between the strength of the attraction force and the strength of the repulsion forces.

The set $F^{\text{on-line}}$ contains the resulting force for each criterion. When the selected alternative is quite different from the current value of the profile, these forces can propose abrupt changes. In order to avoid an erratic behaviour, the parameter mc introduces an upper limit on the amount of change that can be applied on a criterion, expressed as percentage over its range. For instance, a value of 0.5 allows a variation at most of 50% of the range of a criterion (in each iteration). The adaptation mechanism can proceed at a low speed, making small changes to the preferred value for each criterion (if the value of mc is very low), or it can try to make stronger changes in the preference values at each iteration (if the value of mc is large).

At the end of each iteration the selected alternative is stored on the buffer of selections, which is used by the off-line adaptation process explained in the next section.

A high-level representation in pseudo-code of the on-line adaptation process is included in Figure 21.

```

function ON-LINE(
    P, //current profile
    R(an), //selected alternative
    R(a0, ..., an-1), //over-ranked alternatives
    mc, //maximum change of preferences
     $\alpha$ ,  $\beta$ , //adjustment parameters
    n //number of alternatives
)
begin
    changes = { $\emptyset$ };
    positive_change, negative_change = 0;
    k = size(R(an)); //attributes per alternative
    for (j = 0 ; j < k ; j++) do
        // evaluation of the attraction force
        positive_change = Fs(P, R(an), j,  $\alpha$ );
        // evaluation of the repulsion force
        negative_change = Fo(P, R(a0, ..., an-1), j,  $\alpha$ );
        // evaluation of the resulting force
        changes[j] = negative_change + ( $\beta * n * \text{positive\_change}$ );
        // normalization of the proposed change
        changes[j] = normalization(changes[j], j, mc);
    end for;
    return(changes);
    
```

Figure 21. On-line adaptation process

It can be noticed that the techniques designed for the on-line stage fail at detecting user trends over time since they only have information of a single selection. The *off-line adaptation* aims to solve this issue by processing stored information from several user interactions.

4.2.2 Off-line adaptation process

As pointed out before, the *off-line adaptation* process gathers information from several user interactions. This technique allows considering changes in the profile that have a higher reliability than those proposed by the on-line adaptation process, because they are supported by a larger set of data.

The off-line adaptation process can be triggered in two ways: the first one evaluates the user choices, while the second one analyses the over ranked alternatives discarded by the user in several iterations. The possibility of running the off-line process (in any of its two possible forms) is checked after each recommendation.

In the first case, the system has collected some alternatives selected by the user in several recommendation steps. The historical information is used in this off-line adaptation process just when a certain minimum amount of selections have been gathered (rs). The idea is to calculate the attraction forces ($F_i^{s'}$) exerted by each of the stored selected alternatives over the values stored in the profile, using an adaptation of the Eq.(4.1) used in the on-line process:

$$F^{s'}\left(P, \{s^1, \dots, s^{rs}\}, j, \alpha\right) = range(c_j) \sum_{i=1}^{rs} \frac{1}{|s_j^i - P_j|^\alpha} \cdot \frac{s_j^i - P_j}{|s_j^i - P_j|} \quad (4.4)$$

The inputs of this equation are the profile P , the buffer of past selections $\{s^1 \dots s^{rs}\}$, the criterion to evaluate j , and the strength-adjusting parameter α . Again, in this expression it is assumed that $\forall i s_j^i \neq P_j$, where s_j^i is the value of the j -th criterion in the i -th selected alternative. The alternatives in which the value of the criterion j coincides with the preferred value for that criterion in the profile do not participate in this addition. As a result of this evaluation, the following set of attraction forces is obtained:

$$F^{\text{off-line attraction}} = \left\{ F_1^{\text{off-line attraction}}, \dots, F_k^{\text{off-line attraction}} \right\}, \quad (4.5)$$

where $F_j^{\text{off-line attraction}} = F^{s'}\left(P, \{s^1, \dots, s^{rs}\}, j, \alpha\right)$.

The second kind of off-line adaptation process evaluates the set of over ranked alternatives that have been collected through several iterations and which were not used in the on-line adaptation process. This occurs when the on-line process does not have enough over ranked alternatives (mo) in a single recommendation.

When the stored over ranked alternatives reach a certain number (which is the mo parameter used in the on-line adaptation process), the off-line adaptation process is triggered to calculate the repulsion forces over the profile values exerted by those alternatives (F^o), which are calculated using Eq.(4.2). As a result, the following set of forces is obtained:

$$F^{\text{off-line}_{\text{repulsion}}} = \left\{ F_1^{\text{off-line}_{\text{repulsion}}}, \dots, F_k^{\text{off-line}_{\text{repulsion}}} \right\}, \quad (4.6)$$

where $F_j^{\text{off-line}_{\text{repulsion}}} = F^o \left(P, \{o^1, \dots, o^{m_o}\}, j, \alpha \right)$.

It is important to note that the two possible modes of execution of the off-line process (calculation of attraction or repulsion forces) are independent between them. In each iteration, depending on the number of selected and over ranked alternatives that have been accumulated, both of them can be applied, only one of them, or none of them.

In the same way than in the on-line process, before returning the results, the forces are decreased taking into account the maximum permitted percentage of change in one criterion per iteration (parameter mc), avoiding problematic spikes in the adaptation of the preference values.

The high-level representation in pseudo-code of this adaptation process is depicted in Figure 22.

```

function OFF-LINE(
    t, //type of execution (attraction/repulsion)
    P, //current profile
    R(a0, ..., as), //set of stored alternatives
    mc, //maximum change of preferences
    α, //adjustment parameter
)
begin
    k=size(R(a0)); //attributes per alternative
    for (j=0 ; j<k ; j++) do
        if ( isAttraction(t) ) then
            changes[j]=Fs'(P, R(a0, ..., as), j, α);
        else if ( isRepulsion(t) ) then
            changes[j]=Fo(P, R(a0, ..., as), j, α);
        end if;
        changes[j]=normalization(changes[j], j, mc);
    end for;
    return(changes);
end;

```

Figure 22. Off-line adaptation process

The two possibilities of execution of this function are the following:

- If the system has accumulated enough alternatives selected by the user, the first parameter (t) indicates that the attraction forces have to be computed, and the third parameter (R) contains the set of previously selected alternatives. In this case Eq.(4.5) is used to calculate the attraction forces to be applied to the preferred value of each criterion.
- If the system has accumulated enough over ranked alternatives, the first parameter indicates that the repulsion forces have to be computed, and the third parameter contains the set of accumulated over ranked alternatives. Eq.(4.6) is used in this case to obtain the repulsion forces to be applied to the preferred value of each criterion.

4.2.3 Evaluation

The previous sections have presented two algorithms aimed at adapting dynamically and automatically the preferences of the user in the case of numerical criteria, in domains in which the system can monitor a set of continuous decisions made by the user. It can be argued that there is a wide range of decision problems in which users have to make selections in a periodic basis (*e.g.*, the system could learn the musical tastes of the user from his/her continuous selection of songs, his/her preferences on movies from the daily evening selection of a movie to watch on TV, or the kind of news in which he/she is interested from the selection of the news read every morning).

This section describes how we have defined an automatic procedure for the evaluation of the adaptation algorithms, and presents the results obtained in a specific case study. In particular, we detail the analysis that has been made of the possible values for the adjustment parameters α and β , which has permitted to detect the most appropriate values for them.

The evaluation has been conducted in the tourism domain, where the user asks for a recommendation of a tourist destination, based on a certain set of numerical criteria. Figure 23 shows a high-level representation in pseudo-code of the algorithm used to evaluate the adaptation processes. The parameter I is the *ideal profile*, which contains the real preferences of the user. The aim of the adaptation processes is to move the values in the current profile towards those in this ideal profile. The evaluation algorithm simulates *MaxIter* recommendations of the system, and n different alternatives are taken into account in each step. The information about the ideal profile is used in each step to calculate automatically which of the n alternatives would have been chosen by the user.

A set of 1500 alternatives has been used for the purpose of this evaluation. The set of criteria used to define the travel destinations is composed by five attributes (see Table 13): the population density, the average temperature over the year, the distance to the nearest airport, the altitude over the sea level and the average hotel price per night. The whole set of alternatives has been created by generating 1500 alternatives with random values for each criterion, respecting the ranges indicated in Table 13.

The measure used to evaluate and compare results in the different tests conducted in this phase is a *distance measure* that calculates the difference between the ideal profile and the profile that is being learnt or adapted. This is calculated with the following function:

$$\text{dist}(I, P) = \frac{1}{k} \sum_{i=1}^k \frac{|I_i - P_i|}{\text{range}(c_i)}, \quad (4.7)$$

where I_i and P_i are the values of maximum preference (v_{pref}) for the criterion or attribute c_i in the ideal and adapted profiles, respectively, and k is the number of criteria.

```

ADAPTATION-ALGORITHM-EVALUATION(
     $A(a^0, \dots, a^r)$ , //corpus of alternatives
     $mo$ , //minimum number of over ranked alternatives
     $rs$ , //required number of selections
     $\alpha, \beta$ , //adjustment parameters
     $mc$ , //maximum change over profile preferences
     $MaxIter$ , //iterations to simulate
     $I$ , //ideal profile
     $n$  //alternatives per iteration
)
begin
     $P$ =initial-profile();
     $iter, beg, end = 0$ ;
     $overRanked, selected, changes = \{\emptyset\}$ ;
    while (  $iter < MaxIter$  ) do
         $beg = iter * n$ ;
         $end = (iter + 1) * n - 1$ ;
         $R(a^0, \dots, a^{n-1}) = \text{rating-and-ranking}(P, A(a^{beg}, \dots, a^{end}))$ ;
         $u = \text{calculate-ideal-selection}(R, I)$ ;
         $selected = selected + R(a^u)$ ;
        //triggering of the on-line process
        if (  $u < mo$  ) then
             $overRanked = overRanked \cup R(a^0, \dots, a^{u-1})$ ;
        else
             $changes = \text{on-line}(P, R(a^u), R(a^0, \dots, a^{u-1}), mc, \alpha, \beta, n)$ ;
             $\text{apply-changes}(P, changes)$ ;
        end if;
        //off-line process first way of execution
        if (  $\text{size}(overRanked) \geq mo$  ) then
             $changes = \text{off-line}(\text{"repulsion"}, P, overRanked, mc, \alpha)$ ;
             $\text{apply-changes}(P, changes)$ ;
             $overRanked = \{\emptyset\}$ ;
        end if;
        //offline process second way of execution
        if (  $\text{size}(selected) \geq rs$  ) then
             $changes = \text{off-line}(\text{"attraction"}, P, selected, mc, \alpha)$ ;
             $\text{apply-changes}(P, changes)$ ;
             $selected = \{\emptyset\}$ ;
        end if;
         $iter = iter + 1$ ;
    end while;
end;

```

Figure 23. Evaluation of the adaptation algorithms

Table 13. Set of criteria of the analysed alternatives

Criterion	Minimum value	Maximum value	Range
Population density	1 person per km ²	1500 people per km ²	1499 p/km ²
Average temperature	-5° C	25° C	30° C
Distance to the nearest airport	1 km	50 km	49 km
Altitude	0 m	1000 m	1000 m
Average accommodation price	20 €	200 €	180 €

The evaluation explained in this section is divided in three parts: recommendation and adaptation, adjustment of the algorithm parameters and algorithm testing. At first, the process of evaluation and rating of the alternatives at each step of the algorithm is described, including how the user profile is updated according to the (simulated) user selections. Then, an explanation of how the best values for the main parameters of the adaptation algorithms (α and β) to maximize their performance have been determined, is given. Finally, the last part includes the results obtained in three test cases: when the user preferences do not change over time, when the user preferences change randomly over time, and when the user preferences change gradually over time.

Recommendation and adaptation

The evaluation process implies the iterative repetition of the recommendation process several times. To perform these iterations, the initial set of 1500 alternatives was divided in blocks of 15 alternatives (parameter n in Figure 23), producing 100 recommendation problems.

The selection of the value of the parameter n depends on the problem itself. With a relatively low number of alternatives for the user, the available information to estimate his/her behaviour is quite low and the learning curve increases, since the online adaptation process cannot be executed and the whole adaptation depends on the offline one. However, changes applied on the user profile are more reliable since more historical information is considered. On the other hand, when a large number of alternatives is proposed to the user at each iteration, the online adaptation algorithm makes changes in the profile more often, though they are not as reliable as the ones made through the offline process. The selected value is a compromise among these two scenarios.

As it is shown in the pseudo-code in Figure 23, an initial profile should be created. This is an important aspect of the tests and, depending on the initial value, the performance of the algorithm changes. All the results of this section show the average performance of the adaptation algorithm considering three different random initial profiles.

Then, the next stage evaluates the first set of alternatives. As the profile can be seen as another alternative (since in this case it is just a list of values of maximum preference), the same Eq.(4.7) used to calculate the distance between profiles can be used to calculate the distance between a profile and an alternative. Then, the list of 15 alternatives can be ordered according to their individual distance to the current profile.

After the rating and ranking step, a user interacting with the platform would select his/her favourite alternative. In the evaluation that step has been simulated by considering an ideal profile created manually, which is the profile the current profile is wanted to tend to. The selection of the user

is taken to be the alternative with a lowest distance to that ideal profile. The alternatives that were better ranked than this one (those in lower positions) form the set of over ranked alternatives. That piece of information, along with the selected alternative, forms the input data required for the adaptation process.

When the selected and the over ranked alternatives have been identified, the adaptation step takes that information as input in order to decide which changes have to be made in the profile, as explained before.

After the block of 15 alternatives has been evaluated and the relevance feedback extracted, the adaptation step calculates, using Eq.(4.3), the forces used to adapt the profile in the on-line process. If it is necessary to run the off-line adaptation process, the adaptation forces are obtained with Eq.(4.5) and Eq.(4.6).

When the stored profile has been modified taking into account the results of the adaptation processes, the evaluation process obtains the next block of alternatives and starts again the recommendation and adaptation process.

Analysis of the parameters α and β

The influence in the adaptation process of the two main parameters that affect the performance of the recommendation process (α and β) has been analysed. A set of different values has been tested for each parameter to find out the most appropriate ones in this case study.

The parameter α adjusts the strength of the attraction and repulsion forces depending on the difference between the current value of the profile and the value with which it is compared (the selection made by the user or an over ranked alternative, see Eqs.(4.1) and (4.2)).

The behaviour of the function $f = 1/x^\alpha$ depends on x , which is the difference between two values. If x is low (those values are quite similar), taking α greater than 1, the resulting value is quite high, and taking α lower than 1, the final result is too small. On the contrary, if x is high, taking α greater than 1, the obtained result is too small whereas with α lower than 1, the results increase.

The idea is that the resulting attraction/repulsion force should be relatively small when the compared values are similar. Then, values of α greater than 1 are discarded, but the performance with values lower than 1 should be analysed accurately.

Figure 24 shows the behaviour of the adaptation algorithm comparing the distance (measured used Eq.(4.7)) between the user profile and the ideal profile in the iterative adaptation process for different values of α . In all tests, values greater than 1 are not accurate enough to achieve good changes in the profile, whereas values lower than 1 permit the current profile to evolve more precisely towards the ideal profile. Particularly, $\alpha = 3/4$ (as highlighted in Figure 24) offers the best compromise during the adaptation process.

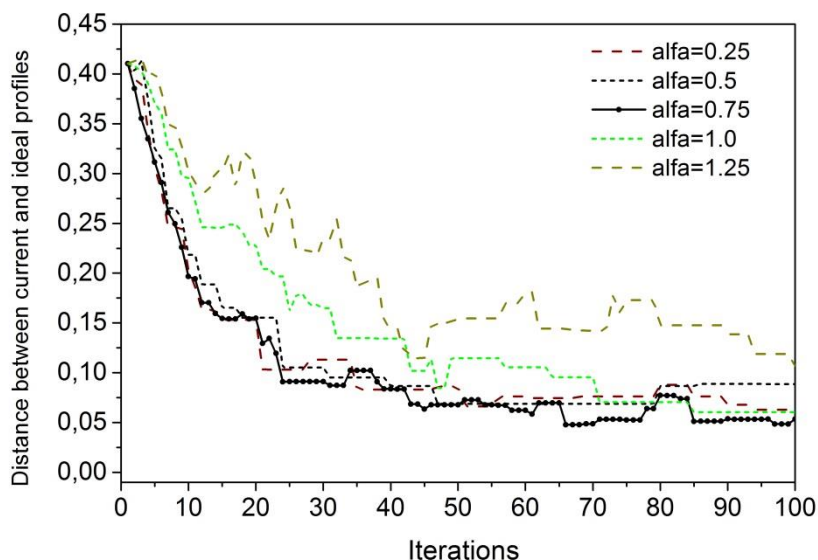


Figure 24. Influence of the parameter α

The parameter β is used in Eq.(4.3) to tune the importance of the selected alternative in relation with the over ranked ones. This parameter was introduced to normalize the importance of the attraction force obtained from the user selection with respect to the repulsion force obtained from the evaluation of the over-ranked set.

Figure 25 compares the performance of the adaptation algorithm taking into account different values of β . Although there are not significant differences among them, the ratio $\beta = 7/4$ offers the best results. It gives more importance to the selection made by the user, but the final change is complemented with the information provided by the analysis of the over-ranked alternatives.

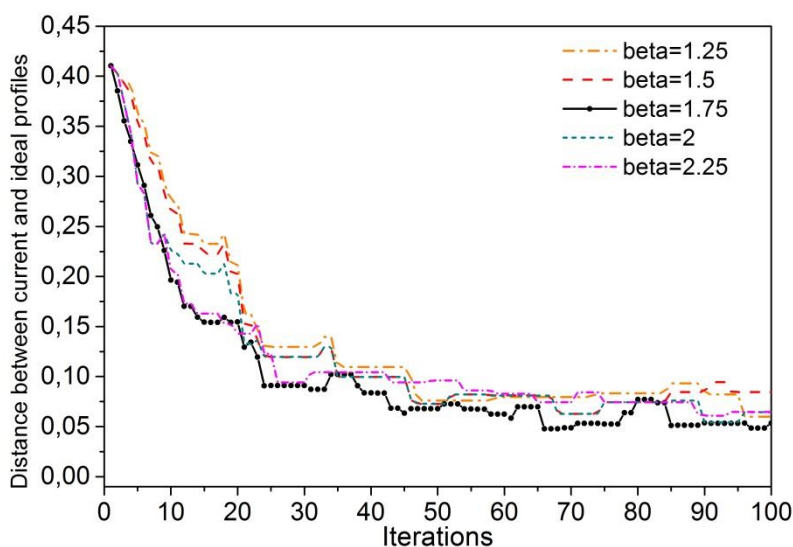


Figure 25. Influence of the parameter β

After choosing the best values for α and β , three kinds of tests have been conducted in order to check the performance of the user profile adaptation algorithm in different scenarios. The first one consists in an

evaluation of the learning process in the case in which the user profile to be learnt is static, the second one introduces random changes in the ideal profile through time, and the third one simulates a user who changes gradually his/her interests about some attributes while using the recommender system.

Evaluation of the learning process

In this first test it is assumed that the ideal profile that the system aims to learn is static (it does not change during the learning process), meaning that the user personal preferences are the same in the first iteration and in the last one. The analysis of the results obtained in this first evaluation has been conducted from three points of view. First, an analysis of the evolution of the distances between the ideal and the adapted profile is made, using only the on-line process or both the on-line and off-line processes. Afterwards, an evaluation of the performance of the RS has been performed, studying the position of the selected alternative on the set of sorted alternatives. Finally, a study on the evolution of the distances between each criterion value in the profile and its ideal one is explained. Five tests with five different random initial profiles have been conducted in order to test our approach and the results included in this section are the average of said five tests.

The graphical representation in Figure 26 shows the performance of the adaptation algorithm, depicting the distance between the ideal profile (the profile that we aim to reach) and the adapted profile (the initial randomly created profile which is modified through the adaptation process).

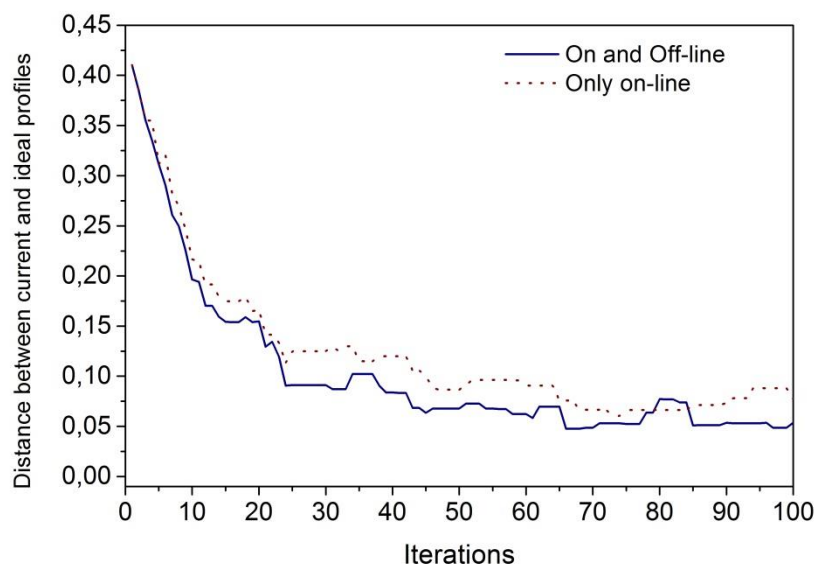


Figure 26. Distance between the ideal and the adapted profiles using the adaptation processes

To understand the meaning of the distance values in this figure, it is necessary to observe that the initial distance of 0.41 means that the preferred value in the profile for each criterion is, in average, at a distance to the ideal value of 41% of the range of the domain of that criterion. This means that if the domain of a certain criterion is 100 units, the initial distance between the profile and the ideal value is 41 units.

Regarding these results, it can be said that it is better to use both the on-line and the off-line processes. In this case, when both of them are used, it can be seen that, from approximately the 40th iteration, the distance between the current profile and the ideal one is quite stable around 0.05. In other words, each adapted value in the profile is, in average, at a distance of only 5% of the domain range to its corresponding ideal value. However, if only the on-line adaptation process is considered, the distance is the double, around 0.10.

It is also worth mentioning that the best results are obtained from the 50th iteration and no significant improvement in the distances is experienced from this point. It can also be observed that in the first 10 iterations the distance already decreases around 50% from the initial value. Thus, the number of iterations needed to tend towards the right profile is quite small.

Figure 27 shows the position of the user selection in the ranking performed by the RS. It can be seen how in the first interactions the selected alternative is not very well ranked by the RS. When observing Figure 26 it was said that the 50th interaction was the point from which the profile was already well adapted. This fact can also be noticed in Figure 27 by observing that from this point the user selection is almost always among the first three alternatives. In fact, the best option is ranked in the first place in most of the iterations.

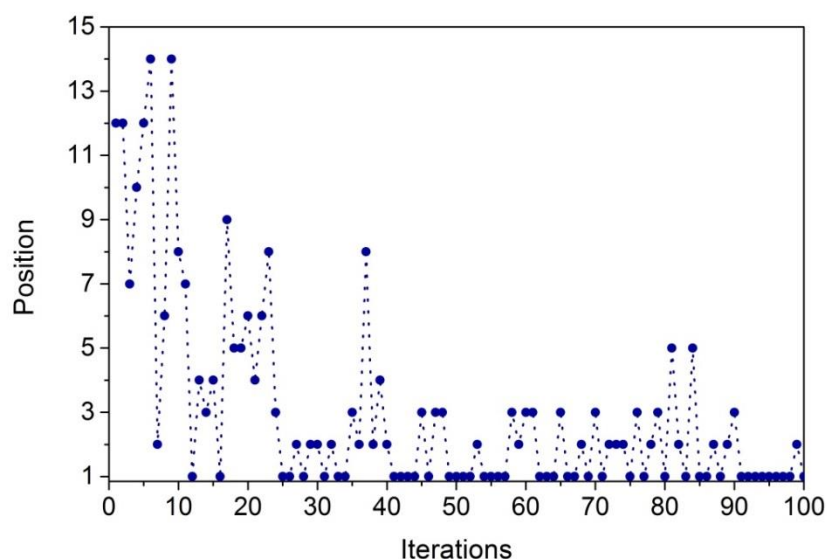


Figure 27. Position of the user’s favourite option in the set of ordered alternatives

Figure 26 shows the evolution of the average distance between the current and the ideal profile. A more detailed analysis can be made showing the evolution of each criterion separately. Figure 28 shows the evolution of the preference values using the whole adaptation mechanism (on-line and off-line modes).

Considering the initial value, two groups of criteria can be observed in the graphic. The first one is composed by “population density”, “average temperature” and “average hotel price”, which have initial values quite similar to the ideal ones. The second one contains the criteria “nearest airport distance” and “altitude”, which have a high initial distance (0.6 and

0.7, respectively) to their ideal respective values. Through the sequence of iterations it can be observed that the criteria in the first set reduce their respective distances to the ideal values slowly. The most dramatic improvement of distances can be observed in the evolution of the second group of criteria in the first 50 iterations, reaching values of 0.02 and 0.12 respectively.

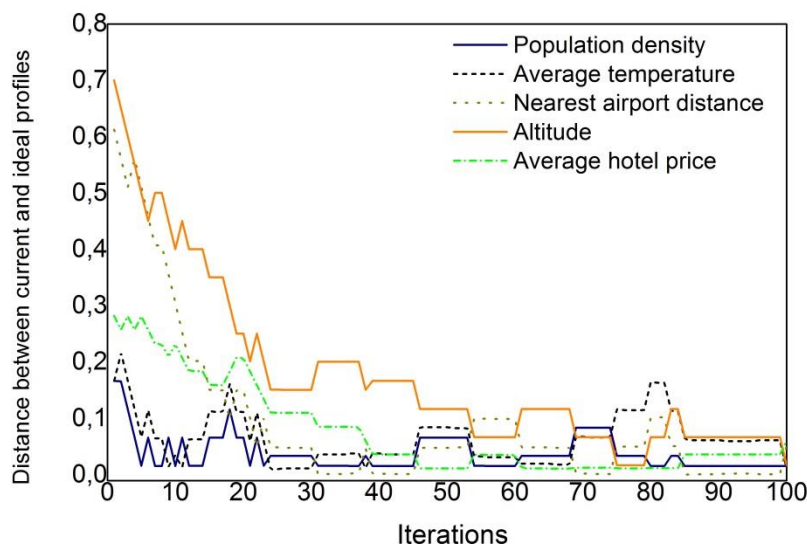


Figure 28. Evolution of the distances for each criterion

As a final note, it can also be noticed that the results show that the adaptation algorithms work correctly regardless of the magnitude of the attributes range, as they do not make absolute changes to the preferences but relative ones.

Evaluation considering random dynamic changes in the ideal profile

This second test has been conducted with the objective of observing how the system reacts to random changes in the ideal profile, simulating a user that is changing very often his/her preferences. The analysis takes into account three parameters: the number of iterations between changes in the profile, the number of attributes in which preference values are changed and the strength of that change. Three different values have been considered for each parameter (giving a total of 27 different tests):

- Time between changes: 1, 5 or 10 iterations.
- Number of attributes to which preference changes are applied: 1, 2 or 5 attributes.
- Strength of the change: 5, 10 or 20 % of the numerical domain of the attribute.

The selection of the attributes that change is made randomly in each iteration, as well as the direction of the change (positive or negative).

Tables 14, 15 and 16 show the results obtained in terms of “Distance between the ideal and the current profile”, “Percentage of iterations in which the selected alternative appears among the first three positions” and “Average position of the selected alternative”. The last two measures are calculated from the 25th iteration to the last. The results shown in the tables

are calculated as the average of 10 different adaptation tests generated with ten different initial profiles using the same changing ideal profile. Each column represents a test in which the preference on 1, 2 or 5 attributes is changed in a 5, 10 or 20 percentage.

Table 14. Preference changes at each iteration (average of 10 tests)

	1 attribute			2 attributes			5 attributes		
	5%	10%	20%	5%	10%	20%	5%	10%	20%
Final distance (user profile, ideal profile)	0.075	0.126	0.183	0.088	0.175	0.209	0.124	0.194	0.272
% of selections among first 3 positions	92.20	75.30	37.70	94.80	58.40	23.40	74.00	32.50	3.90
Average position of the selection	2.084	2.547	3.468	2.249	2.981	4.017	2.723	3.581	5.168

Table 15. Preference changes every 5 iterations (average of 10 tests)

	1 attribute			2 attributes			5 attributes		
	5%	10%	20%	5%	10%	20%	5%	10%	20%
Final distance (user profile, ideal profile)	0.058	0.065	0.099	0.067	0.089	0.151	0.063	0.103	0.194
% of selections among first 3 positions	92.20	96.10	80.50	93.50	85.70	55.80	94.80	77.90	31.20
Average position of the selection	1.981	1.877	2.536	1.977	2.312	3.023	2.191	2.557	3.651

Table 16. Preference changes every 10 iterations (average of 10 tests)

	1 attribute			2 attributes			5 attributes		
	5%	10%	20%	5%	10%	20%	5%	10%	20%
Final distance (user profile, ideal profile)	0.052	0.066	0.079	0.053	0.095	0.111	0.073	0.080	0.160
% of selections among first 3 positions	93.50	97.40	94.80	96.10	93.50	80.50	98.70	90.90	49.40
Average position of the selection	1.821	1.851	2.075	1.791	1.964	2.456	1.906	2.194	3.012

The values of the final distance between the learned user profile and the ideal profile are shown in green if they are below 0.10, and in red if they are above 0.20. The worst results for this indicator, as expected, are obtained when the preference value of 2 or 5 attributes is randomly changed by a factor of 20% of the domain range at each iteration (an extremely unrealistic situation).

Concerning the percentage of iterations in which the user selection appears in the first 3 positions of the list of alternatives ranked by the system, it is shown in green if it is above 90%, and in red if it is below 70%. The worst results are obtained when the preferences are changed in every iteration by a factor of 10% or 20%. Even in those cases in which the preferences are changed only every 10 iterations there is a bad percentage (49.4%) if we change the preferences of the five attributes by a 20% factor.

The third evaluation indicator, the average position in the ranked list of options of the selected alternative (the alternative that fits better with the ideal profile) is shown in green if it is below 3, and in red if it is above 5. These results are quite satisfactory, the worst case being the one in which the preferences of the five attributes are changed by a 20% factor at each iteration.

This set of results of 27 exhaustive tests shows that the user profile learning algorithm reacts appropriately to the dynamic changes in the preferences of the user, even if these changes are quite strong and happen quite often.

Evaluation considering a dynamic, gradual and continuous change of the user preferences

The purpose of this test is to show a more realistic example of application of the learning algorithm in a situation in which the user preferences change gradually on time. For the evaluation of this test, let us suppose a case in which the ideal user preferences are initially defined as follows:

- Population density: 250 inhabitants/km²
- Average temperature: 15°C
- Distance to the nearest airport: 25 km
- Altitude: 100m
- Average hotel price: 60€

Let us assume that, as the user grows older, his/her preferences over two of those attributes change gradually: he/she is going to prefer destinations that are near an airport to avoid long transports to the city centre (10 km), and he/she is also going to prefer more expensive hotels (95€). The rest of the preferences remain the same.

Figure 29 and Figure 30 show the evolution of the preference values in the user profile for the attributes *nearest airport distance* (NAD) and *average hotel price* (AHP), respectively. Both figures show the evolution of the preferred value of the analysed criterion in two cases: when the user's preferences change over time and when they remain the same, that is, static at the original value (25km in the first attribute and 60€ in the second one).

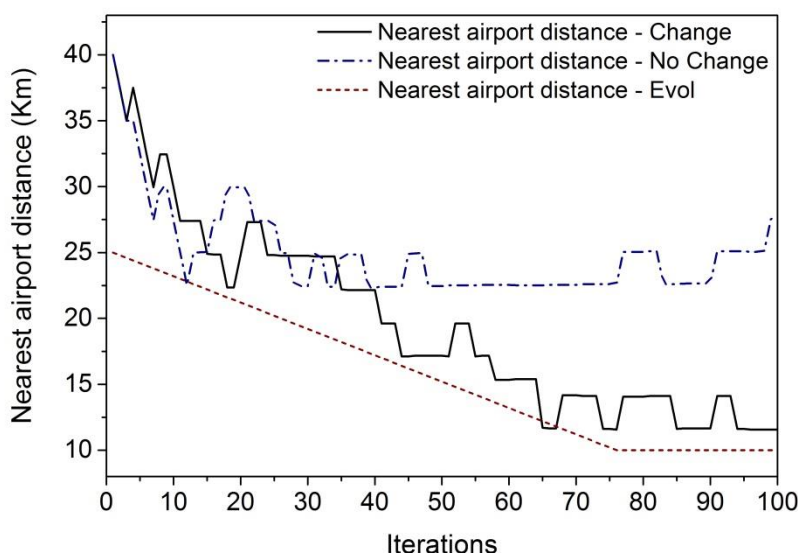


Figure 29. Evolution of the *nearest airport distance* (NAD) preference values

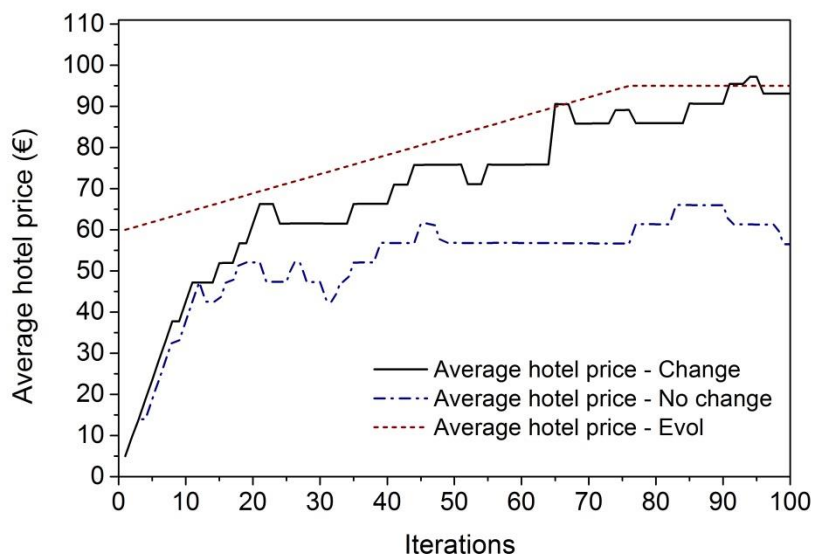


Figure 30. Evolution of the *average hotel price* (AHP) preference values

In the first case, it is supposed that during the first 75 iterations the user preference over the NAD decreases from 25 km to 10 km gradually, and then stays stable till the end. It can be seen how the attribute's preference value in the current profile is adapted accordingly and assimilates that change. The “no change” series shows how the preference value would change if the ideal preference value is not modified during the simulation.

The same occurs for the second case, in which the user's preference value on the attribute AHP increases linearly from 60€ to 95€ in the ideal profile during the first 75 iterations. Here the value of preference in the current profile also increases following that change. The “no change” series shows how the preference value would change if the ideal preference value is 60 during all the simulation.

Although the numeric preference learning approach described in the previous sections provided an adequate way of learning the ideal value of preference over a numeric attribute, it was unable to learn all of the parameters that model the preference function such as the slope or the width, as explained in Chapter 2. This is why this model has been extended to allow the learning of the whole set of parameters that intervene in the preference function.

4.2.4 Preference function learning

The new learning method presented in this subsection relies on historic data about the user selections to approximate the preference function of the numeric attributes to the most adequate one. The whole process of learning the numerical preference function is performed after the system has stored at least ten interactions/selections of the user, since with fewer data the learned function would probably not be accurate enough. Note that with this approach, the function of preference is defined by 5 parameters (left and right slope, left and right width, and value of maximum preference) instead of considering only the preferred value (see Eq.(2.1)).

```

function PREFER-FUNC-ADAPTATION (
     $V(v_0, \dots, v_n)$ , //historic of values of past selections
     $v_{pref}$  //value of maximum preference
     $v_{min}$ , //minimum numeric value
     $v_{max}$  //maximum numeric value
     $ti$ , //trust interval
     $s$  //probability distribution sampling
)
begin
     $B$ =getBestValues( $V, v_{pref}, ti$ );
     $PD$ =calculateProbabilityDistribution( $B, v_{min}, v_{max}, s$ );
     $\Delta\{left, right\}$ =calculateDelta( $PD$ );
     $m\{left, right\}$ =calculateBestSlope( $PD, v_{pref}, \Delta$ );
     $preferenceFunction$ =( $\Delta, m, v_{pref}$ );
return  $preferenceFunction$ ;
end;

```

Figure 31. Preference function learning algorithm

The learning process of the numeric preference function, depicted in Figure 31, basically has the following steps:

- 1) Obtain the historic values in a trust interval.
- 2) Calculate the probability distribution of the values in that interval.
- 3) Calculate the delta and slope values based in that probability distribution.
- 4) Elaborate the new preference function.

Figure 32 graphically represents the three main steps in the numeric preference function learning process.

The first step filters the more reliable values from the historic set of selections by extracting a percentage of the values closer to the value of preference (*trust interval*), for instance 90%. This is a common way to get rid of outlier values that might disturb the learning process.

Then a *probability distribution function*, represented with a histogram, is calculated with the remaining values, as depicted in Figure 32a. The sample or discretization step is a parameter, for instance 10% of the domain range as used in the figure. Delta values are then calculated by observing the width of the probability distribution. For example, if the first domain value with a positive frequency in the histogram is 3, the last one is 56 and the value of higher preference (v_{pref}) is 34, Δ_l would be 31 and Δ_r would be 22.

Afterwards, the algorithm generates preference functions with different combinations of values for the slope values (m) (in the range from 0 to 5 in steps of 0.1), and compares the distance between each preference function and the probability distribution. The distance is calculated by discretizing both function domains in steps of 1% and adding the difference of the function values for those discretized points. The function with the lowest distance determines the best slope. In Figure 32b several functions with different slopes are depicted for both the left and the right sides of the function, the best of them being the ones shown in bold line: $m = 4$ for the left side (m_l) and $m = 0.7$ for the right side (m_r).

Finally, the new preference function is built with the new delta and slope values (Figure 32c). The v_{pref} value is still learnt with the technique explained in the previous sections.

The analysis of the whole preference learning process with this improvement is included in the next chapter, in which it is integrated with the automatic learning of preferences on multi-valued categorical attributes.

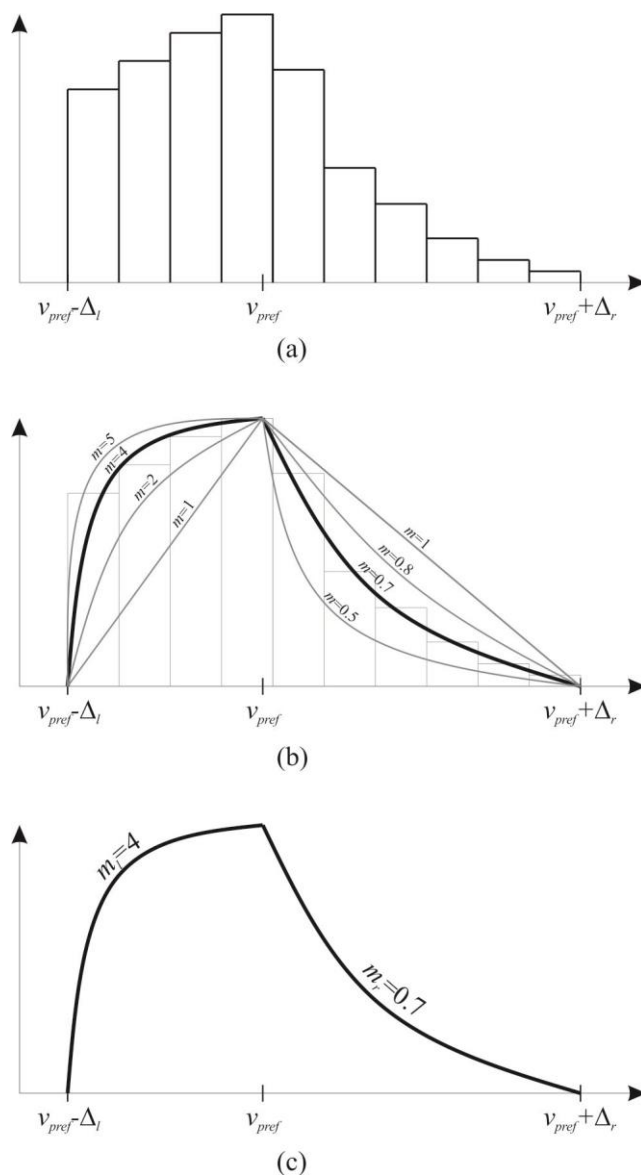


Figure 32. Preference function learning steps

4.3 Preference learning over categorical attributes

The main idea behind preference learning over categorical attributes is to find attribute values repeated among the over ranked alternatives that do not appear on the selection, which will be the candidates for having his/her preference decreased. Similarly, the preference of the attribute values that appear on the selection and do not appear often on the over ranked alternatives is likely to be increased.

Table 17 shows an example of a recommendation result given by the system for a set of seven pieces of news obtained from *The New York Times*, evaluated using the categorical attributes “Desk”, “Section”, “Extension” and “Geographical location”. The first column indicates the global preference score obtained for each news article. For example, the element in the first row is qualified as having a “Very High” affinity with the user profile, whereas the fourth only matches the profile to a “Medium” degree. In this example, the user has selected the sixth news article as the most interesting for him/her. Two pieces of information (relevance feedback) are extracted from the selection made by the user: (1) the alternative selected by the user and (2) the set of alternatives that were ranked above the selected one. The main idea is that the adaptation process must be able to internally find an answer to questions such as, “why are those alternatives better ranked than the one the user really wants?” or “why does the alternative the user really wants not have a good enough score to be ranked in the first place?”.

Table 17. An example set of rated and sorted alternatives

Rank	Desk	Section	Extension	Location	Selection
1 (Very High)	Sport	Wold	Short	USA	
2 (High)	Sport	Sports	Short	Spain	
3 (Medium)	National	Technology	Long	USA	
4 (Medium)	National	Technology	Long	USA	
5 (Low)	Business and Financial	Science	Very long	Germany	
6 (Low)	Foreign	World	Medium	Germany	•
7 (Very Low)	Sport	Business	Very short	USA	

As in the numeric adaptation, the profile adaptation for categorical attributes is also conducted by two processes, depicted in Figure 33.

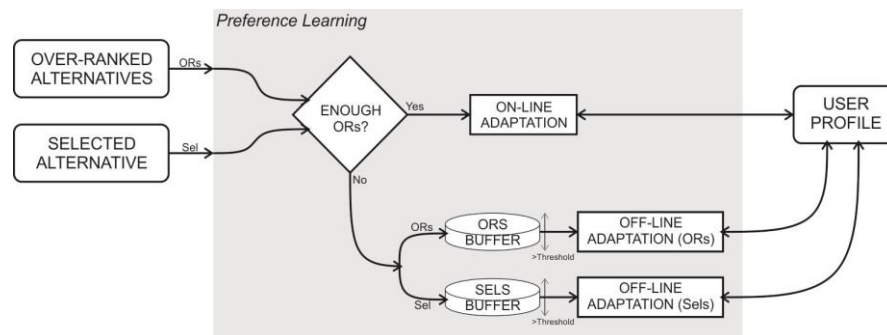


Figure 33. Preference learning process

The first one—called *on-line adaptation*—is executed every time the user asks the system for a recommendation and evaluates the information that can be extracted from the current set of ranked alternatives. The second one—called *off-line adaptation*—is triggered after the recommender system has been used a certain number of times. It considers the information given by the history of the previous rankings of alternatives and the selections made by the user in each case, but considers that information separately.

These adaptation processes modify the user profile by increasing or decreasing the level of preference of certain values of the criteria according to the reasoning mechanism explained below. The algorithm accumulates evidence to update all values, decides which values should be updated, and finally changes them by increasing/decreasing the preference score given the order of the linguistic terms in the set of fuzzy preference terms S . For instance, a value with the preference “High” can be increased to “Very High” or decreased to “Medium”. All these processes will be described in detail throughout the following sections. Table 18 summarizes all the parameters of the adaptation algorithm, providing its identifier and a brief description and showing at which part of the whole process they are used (on-line, off-line, or in the adaptation stage when the final changes to be made are decided).

Table 18. Linguistic preference adaptation process parameters

Id.	Description	On-line	Off-line	Final
t	Percentage of over-ranked alternatives that must have the same value if a characteristic is to be taken into account (e.g., 25%, 50%, 75%)	•	•	
mo	Minimum number of over-ranked alternatives needed to extract characteristics (e.g., 2, 5, 10)	•	•	
h	Minimum number of selections that must be stored before they are evaluated (e.g., 2, 5, 10)		•	
pc	Maximum number of changes in the user profile in one adaptation step (e.g., 1, 3, 5)			•
k	Level of evidence needed to change a preference value. The sign determines if the change is to increase (+) or decrease (-) this value (e.g., +/-2, +/-5, +/-10)			•

4.3.1 On-line adaptation process

The on-line profile adaptation process tries to keep the user profile updated by evaluating each of the user’s selections, without taking into account the previous usage of the system. The main goals of this stage are to decrease the preference of the attribute values that are causing non-desired alternatives to be given high scores and to increase the preference of the attribute values that are important for the user but are not well judged on the basis of the current user profile. The pseudo-code of this process is summarized in Figure 34.

As said before, for each recommendation made by the system, two sources of information are evaluated: the selected alternative, which is the choice made by the user, and the alternatives that were ranked above it. Many conclusions can be derived from this information by extracting *characteristics* from the available data (first loop in Figure 34). A characteristic is a tuple $\langle c_i, v_{ij}, n \rangle$ consisting of the name of one of the attributes (c_i), a value of the attribute (v_{ij}) and the number of times it is

repeated among a concrete set of alternatives (n) (e.g. $\langle \text{"Length"}, \text{"Short"}, 2 \rangle$). These tuples are generated in the second and third loops in Figure 34. A minimum number of alternatives is required, because if characteristics are extracted from a reduced number of alternatives they do not give useful information.

```

function ON-LINE(
     $R(a^n)$ , //selected alternative
     $R(a^0, \dots, a^{n-1})$ , //corpus of sorted over-ranked alternatives
     $t$  //characteristics extraction threshold
)
begin
    evidences= $\emptyset$ ;
    overRankedChars, selectionChars= $\emptyset$ ;
    numAts=size( $R(a^n)$ ); //attributes per alternative
     $M=\max(t \cdot u, 2)$ ;
    for ( $i=1; i < \text{numAts}$ ) do
        overRankedChars=overRankedChars+
            getOverRankedChars( $R(a^n), R(a^0, \dots, a^{n-1}), i, M$ );
        selectionChars=selectionChars+
            getSelectionChars( $R(a^n), R(a^0, \dots, a^{n-1}), i, M$ );
    endfor;
    for ( $i=1; i < \text{size}(\text{overRankedChars})$ ) do
        evid= $\langle$ attribute( $\text{overRankedChars}(i)$ ),
            value( $\text{overRankedChars}(i)$ ),
            repetition( $\text{overRankedChars}(i)$ ), DECREASE $\rangle$ ;
        evidences=evidences $\cup$ evid;
    end for;
    for ( $i=1; i < \text{size}(\text{selectionChars})$ ) do
        evid= $\langle$ attribute( $\text{selectionChars}(i)$ ),
            value( $\text{selectionChars}(i)$ ),
            repetition( $\text{selectionChars}(i)$ ), INCREASE $\rangle$ ;
        evidences=evidences $\cup$ evid;
    end for;
    return(evidences);
end;

```

Figure 34. Pseudo-code of the on-line process

The features extracted from the set of alternatives that were ranked above the user's final selection are referred to as *overranked characteristics*, and they contain elements that were not selected by the user. These characteristics are detected by observing the repetitions of the values in this set, but only when it has enough elements. These two conditions are modelled using two parameters: t and mo . The first parameter helps to identify relevant characteristics when a value that does not appear in the user selection is repeated in the over-ranked alternatives in a percentage over t . When the over-ranked set of alternatives contains only a few elements, erroneous evaluations can be produced. The parameter mo sets a minimum number of elements for this stage to be performed.

For instance, Table 17 shows an example in which the user selects the sixth alternative. Setting a threshold $t = 40\%$, the tuple $\langle \text{"Length"}, \text{"Short"}, 2 \rangle$ is a correct over-ranked characteristic, the attribute name is “Length”, “Short” is one of the possible attribute values, and 2 is the number of times it appears within the five over-ranked alternatives ($2/5 \geq 40\%$). Note that $\langle \text{"Length"}, \text{"Very long"}, 1 \rangle$ is not an over-ranked characteristic to be considered because the value “Very long” only appears once in the five over-ranked alternatives ($1/5 < 40\%$).

Over-ranked characteristics are used to decrease the level of preference of the attribute values in the over-ranked set. The intensity of the decrease is regulated by the number of repetitions: if a characteristic is repeated many times, there is more reason to decrease the preference of the attribute value.

The features extracted from the user’s final selection that do not appear in the set of over-ranked alternatives more than a given number of times are called *selection characteristics*. In this case, the repetition threshold t is used quite differently from the situation in which over-ranked characteristics are extracted. The adaptation process will only consider those selection characteristics that appear among the over ranked alternatives in a percentage lower than this threshold. Following the example in Table 17 and considering a threshold t of 40%, the tuples $\langle \text{"Length"}, \text{"Medium"}, 0 \rangle$ and $\langle \text{"Section"}, \text{"World"}, 1 \rangle$ are correct selection characteristics. In a similar procedure, selection characteristics are used to increase the level of preference of the attribute values indicated by the characteristics.

The less the value appears among the over ranked characteristics, the greater the intensity of the increase. In other words, in the example above, the preference of the value “Medium” of the attribute “Length” should be increased because it is not among the over-ranked alternatives, but the value “World” of the attribute “Section” should not, because it appears once.

4.3.2 Off-line adaptation process

The on-line adaptation process can give an immediate response to every interaction with the recommender system. However, there are cases in which the information extracted from a single interaction does not provide enough evidence to make changes to the profile. In these cases, data from past recommendations are stored and accumulated to be analysed later. The off-line adaptation process manages over ranked characteristics and selections differently, as explained below (see the pseudo-code in Figure 35).

Notice that in the on-line adaptation the parameter mo defines the minimum number of over-ranked alternatives that are required before characteristics can be extracted. However, this causes the following problem: when the profile is quite good (i.e. it accurately represents the preferences of the user’s ideal profile, I), the user selection will be among the first ranked alternatives, and there will not be enough over-ranked alternatives to analyse. This situation means that the system is unable to determine which preferences it must increase or decrease to reach the ideal

profile. One solution to this problem may be to store these small pieces of data and evaluate them as a whole when there is enough information. Because this information comes from different recommendations, it is not appropriate to compare over-ranked alternatives with the selected alternatives as in the on-line process. However, some evidence for preference decrease can still be extracted by evaluating the over-ranked alternatives separately (first loop in Figure 35), and evidence for preference increase can be extracted by evaluating the selected alternatives separately (second loop in Figure 35).

```

function OFF-LINE(
    selected, //set of previously selected alternatives
    overRanked, //set of over ranked alternatives
    t, //characteristics extraction threshold
    mo, //minimum number of over-ranked alternatives
    h //number of selections
)
begin
    evidences=∅;
    M=max(t:size(overRanked)),2);
    if (size(overRanked) ≥ mo) then
        for (i=1; i < size(overRanked)) do
            evid=⟨attribute(overRanked(i)),
                value(overRanked(i)),
                repetition(overRanked(i)), DECREASE⟩;
            evidences=evidences∪evid;
        end for;
    end if;
    if (size(selected) ≥ h) then
        for (i=1; i < size(selected)) do
            evid=⟨attribute(selected(i)),
                value(selected(i)),
                repetition(selected(i)), INCREASE⟩;
            evidences=evidences∪evid;
        end for;
    end if;
    return(evidences);
end;
    
```

Figure 35. Pseudo-code of the off-line adaptation process

When the system faces cases in which the number of over-ranked alternatives is not large enough for reliable characteristics to be extracted (parameter *mo*), it stores the small number of over-ranked alternatives in a temporary buffer. After several iterations in which the number of over-ranked alternatives has been insufficient for evaluation, the system will have recorded enough alternatives to start evaluating them. When there are enough saved over-ranked alternatives, over-ranked characteristics are extracted from the set of accumulated alternatives. These characteristics are used to decrease the preferences in the same way as in the on-line process:

characteristics with a repetition value that is above the defined threshold are decreased. After the stored over-ranked alternatives have been evaluated, they are erased from the temporary buffer. The number of over ranked alternatives to be stored before they are evaluated is determined by the parameter mo used in the on-line process.

On the other hand, user selections are stored and, after a certain number of choices have been made, they are evaluated. The number of selections needed for an evaluation is described by the parameter h . By extracting selection characteristics from the set of stored selections, the preference of the most repeated attribute values can be increased because their repeated selection indicates that the user is interested in them. After the stored selections have been evaluated, they are removed from the buffer.

4.3.3 Adaptation mechanism

After a single recommendation and the subsequent selection by the user, many characteristics can be found, and the system can make deductions about numerous changes to be made to the profile. Although many of those changes — also known as adaptations — are deduced using the reasoning techniques defined above, they can be incorrect.

This problem has been addressed by restricting the number of adaptations that can be made to the profile every time the process is executed. The parameter pc limits the number of increases and decreases that can be made to a profile per adaptation step (a value of $pc = 2$ allows a total of 4 changes: 2 increases and 2 decreases). When many adaptations are being considered, only the most evident ones are performed. For decreases, the level of evidence is measured by the number of times the value of the characteristic is repeated and the characteristics with most repetitions are decreased. On the other hand, increases are measured by counting the number of over-ranked alternatives that do not have the value of the characteristic and the ones with the highest number are increased. This mechanism reduces the errors that can be made by wrongly increasing/decreasing the preferences, although it may be inefficient when the levels of evidence are the same in many possible adaptations, including the wrong ones. The algorithm is presented in Figure 36.

The approach studied to solve this problem involves a signed counter that regulates the final increase/decrease of the preference associated to a value in the profile. When the adaptation process detects evidence for a possible adaptation, it does not directly apply the change. First, the system increases/reduces a counter, initially set to 0, which is associated to each attribute value (for example, to the value “Germany” of the attribute “Geographic location”). The counter is increased or decreased according to the level of evidence of the extracted characteristic, as explained above. When this counter reaches a certain value defined by the parameter k , the increase/reduction is finally performed in the profile, and the counter is reset to 0. For instance, considering $t = \pm 5$ and following the example in Table 17, the system decides that the preference degree for the value “Germany” of the attribute “Geographic location” needs to be increased. However, its counter is only increased by four units because the value “Germany” appears once among the five over-ranked alternatives. Therefore, the counter does not reach the minimum value required for the

preference degree of “Germany” to be increased in the user profile, and further positive evidence will be necessary before it is changed. This mechanism was motivated by the observation that the great majority of erroneous adaptations have low levels of evidence. With this method, several consecutive erroneous pieces of evidences would have to be observed before an incorrect change is made to the profile.

```

function ADAPTATION-ALGORITHM(
    E(a0,...,amax), //corpus of alternatives
    t, //characteristics extraction threshold
    mo, //minimum number of over-ranked alternatives
    h, // number of selections
    pc, //maximum preferences changes
    k, //level of evidence
    ai, //alternatives per iteration
    MaxIter, //iterations to simulate
    I //ideal profile
)
begin
    P=initial-profile();
    iter, beg, end=0;
    overRanked, selected, evidences=∅;
    while (iter < MaxIter) do
        beg=iter·ai;
        end=(iter+1)·ai-1;
        R(a'0,...,a'ai-1)=rating-and-ranking(P,E(abeg,...,aend));
        u=calculate-ideal-selection(R,I);
        selected=selected ∪ R(a'u);
        if (u < mo) then
            overRanked=overRanked ∪ R(a'0,...,a'u-1);
        else
            evidences=on-line(R(a'u), R(a'0,...,a'u-1), t);
        end if;
        evidences=evidences+off-line(selected, overRanked, t, mo, h);
        if (size(selected) ≥ h) then selected=∅;
        if (size(overRanked) ≥ mo) then overRanked=∅;
        P=adaptation(P, evidences, pc, k);
        iter=iter+1;
    end while;
end;

```

Figure 36. Pseudo-code of the simulator

4.3.4 Evaluation

This subsection presents the evaluation of the proposed adaptation algorithm with real data taken from news published in *The New York Times*. First, the Web-based platform that has been designed and implemented to automatically test the adaptation process is described. The subsection below describes the criteria considered for each news article and the possible values for each criterion. Then an explanation of how the difference between the current profile and the ideal profile we aim to learn is computed after each iteration of the adaptation process is included.

Finally, a comprehensive collection of tests that have been performed to study the influence of each of the main parameters of the user profile adaptation algorithm is presented.

Web-based platform

A web platform has been implemented to test and evaluate the recommender framework presented in this section. It permits to define a problem (the set of criteria and the aggregation and adaptation parameters), an initial profile, and the ideal profile that the system aims to learn; a corpus of alternatives for the problem to be generated (or uploaded); the interaction of the user with the system to be simulated; and the evolution of the user profile to be displayed. Access to the platform is regulated by a username and a password, which allows multiple users with different profiles for each recommendation domain. This approach makes it possible to analyse the variability of the results under different circumstances in the aggregation and adaptation stages. Users can upload data files with information about the alternatives that must be analysed before a decision can be taken. The platform stores the information in a database, which allows users to define a decision-making problem on the basis of these alternatives.

The design of the adaptation algorithm shown in Figure 36 enables the adaptation procedure to be automatically validated by simulating how the system adapts an initial profile with the information extracted from the user selections that are automatically calculated using the ideal profile to be learned. The platform also permits a real user to perform the selection.

Each simulation starts with an initial profile, which may be given by the user or generated randomly. The first task in each iteration is to rate and rank a block R of alternatives. Each alternative is rated by aggregating (as seen in Chapter 3) the preference for the value of each criterion stored in the current profile (P). In this automatic simulation mode, the system selects the alternative that would obtain the best rating with the ideal profile I (i.e., the alternative that, presumably, would be chosen by the user if confronted with the set of alternatives R). The selected alternative and the set of over-ranked alternatives are used to find the evidence in the on-line adaptation step. The evidence is based on the repetition of attribute values and is used to determine whether the preference for a value should be increased or decreased. The off-line stage complements the on-line stage to compose the set of evidences used to adapt the current profile.

Testing domain

Although hundreds of news articles are published every day, users only read those that are most related to their interests (*e.g.*, the news that appears in a particular section of the newspaper, or about a particular football team). Under this scenario we applied the adaptation algorithm to learn the user preferences which were then used by the recommender system to filter the news and present users with only the most relevant items. The corpus of alternatives used in this evaluation consists of 3200 news articles

published in *The New York Times*, which have been obtained using the public *Article Search API*¹.

The attributes that have been taken into account to evaluate these news articles and the possible values for each of them are: the *desk* (internal department of *The New York Times*) that produced the story, the *section* of the newspaper in which the article appears, the *length* of the article and the *geographic location* where the story takes place. The length attribute values (initially numeric) were transformed to categorical values by using the following rules: “Very Short” if it has less than 450 words, “Short” if it has between 450 and 700 words, “Medium” if it has between 700 and 850 words, “Long” if it has between 850 and 1100 words, and “Very Long” if it has more than 1100 words. Possible values for each attribute are depicted in Figure 37.

“The New York Times” news attributes			
Desk	Section	Length	Geographic location
-Sport	-Sports	-Very short	-United States
-Foreign	-Technology	-Short	-Spain
-Business & Financial	-World	-Normal	-France
-National	-Science	-Long	-Germany
-Other desks	-Business	-Very long	-Japan
			-China

Figure 37. News attributes and possible values

Evaluation procedure

A recommender system can be evaluated in terms of precision (how many of the items recommended are interesting for the user) and recall (how many of the items that are of interest for the user are actually recommended) (Morales-del-Castillo et al. 2010; Porcel, Herrera-Viedma 2010; Serrano-Guerrero et al. 2011). In our case, the nature of the system makes it difficult to perform such an evaluation, which requires a domain labelled a priori with the correct selections for a specified user and a corpus of examples to recommend. For this reason, it is proposed to iteratively measure the distance between the current user profile and the ideal profile that the system wants to learn, similarly as it was done in the evaluation of the numeric preference learning processes. Iteratively and automatically, the system selects the alternative that best fits the ideal profile, which is the one that the user would choose. Then the selection and the over ranked alternatives are used by the adaptation processes to evaluate which changes should be made to the current profile.

To compare the results of different tests, several simulations were performed that take three different initial profiles into account but maintain the ideal user profile. The distance between the current profile $P = \{p_1, \dots, p_n\}$ and the ideal profile $I = \{\eta_1, \dots, \eta_n\}$ is calculated in the following way:

$$dist(I, P) = \frac{1}{n} \sum_{j=1}^n \frac{1}{card(c_j)} \sum_{k=1}^{card(c_j)} \left| \frac{CoG(p_j(v_{jk})) - CoG(\eta_j(v_{jk}))}{CoG(s_0) - CoG(s_T)} \right| \quad (4.8)$$

¹ *New York Times* Article Search API: http://developer.nytimes.com/docs/article_search_api (Last accessed: April 26th, 2013).

In this definition, n is the number of criteria, $card(c_j)$ is the cardinality of criterion j , and the functions p_j and η_j return the linguistic preference value of the attribute j for a given value v_{jk} according to the profiles P and I , respectively. Those linguistic preference labels are defuzzified using the center of gravity (CoG, as in Definition 3.5) to calculate a numerical distance value. With this function, we obtain an average of the distances between all pairs of labels. The distance is normalized with the extreme values of the linguistic labels set $S = \{s_i, i \in \{0, \dots, T\}\}$.

The distance is 0 when both profiles are identical. The maximum distance is 1 when all values contained in the user and the ideal profiles have the opposite extreme labels (s_0 and s_T). Moreover, this function is commutative ($dist(P, I) = dist(I, P)$) due to the absolute value of the difference.

All tests were made using the same corpus of 3200 alternatives described above, in blocks of 16 alternatives at each step of the evaluation process, which allowed up to 200 different iterations of the recommender process. Profiles (ideal and initial) were initialized manually (using the attributes indicated previously) using a balanced term set of five balanced terms (“Very low”, “Low”, “Medium”, “High” and “Very high”, see Figure 38) to obtain comparable results.

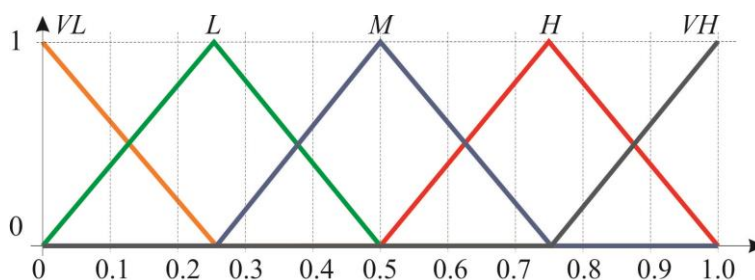


Figure 38. Fuzzy label set with 5 balanced terms

Each of the parameters explained previously (t , mo , h , pc and k) was tested with three different values. Moreover, each of these values was tested with three different initial profiles. All the tests discussed in this section were performed using an extraction characteristic threshold t of 25%, with a level of evidence of $k = \pm 5$ if values are to be allowed to change. A maximum of three changes ($pc = 3$) were allowed in each direction in each iteration, a minimum of five user selections were stored ($h = 5$) and at least five over-ranked alternatives ($mo = 5$) were required to extract characteristics.

Figure 39 compares the performance of the adaptation algorithm using the on-line stage by itself, and also in conjunction with the off-line stage. When both stages were carried out (straight line), the results decreased from an initial distance of 0.62 to 0.2. However, without the off-line stage (dashed line), the final distance was around 0.36.

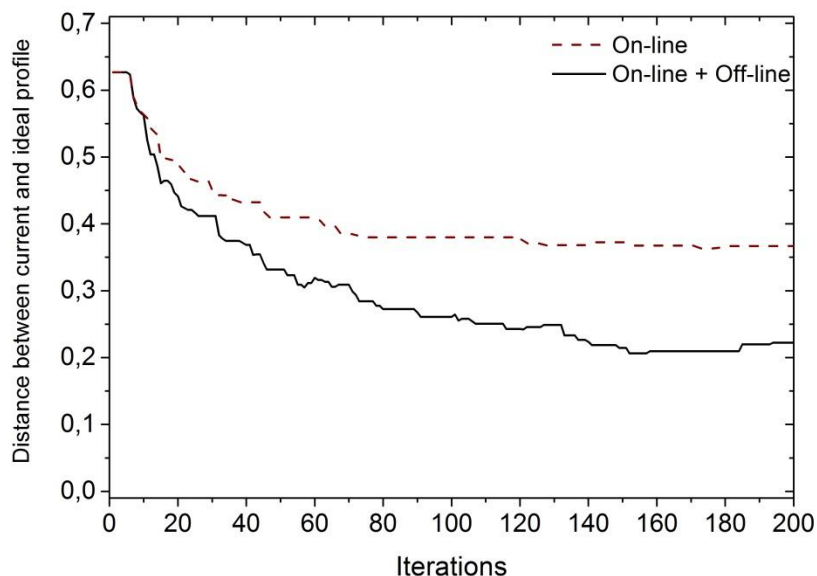


Figure 39. Distances between the current and the ideal profiles

As mentioned above, the results obtained with Eq.(4.8) are normalized between 0 and 1 by dividing them by the maximum difference between the terms in set S , which is $CoG(VL) - CoG(VH) = 0.83$. Notice that as the domain is divided into five labels, the distance between the top of one label and the next is 0.25. A final distance between the user profile and the ideal profile of 0.2 (using both adaptation processes) means that the average distance between the preference values for each criterion value is close to 0.16 units (less than one label away per criterion value). Considering only the on-line process, the final distance between profiles is about 0.37 which represents that the average distance between each of the 21 preference values and the ideal one is close to 0.29 units (a little more than one label away per criterion value).

The results also evolve more quickly when both stages are used, not only the on-line stage. Both processes together propose a more complete set of changes, which improve the performance of the algorithm.

Figure 40 focuses on the study of one simulation and shows the evolution of the distance between the preference labels of the current and ideal profiles, for each attribute value. For instance, if the preference of an attribute value of the current profile is H (“High”) and the ideal is L (“Low”), the distance is 2. The figure compares how many attribute values there are with a difference of 0 (correctly classified), 1, 2, 3 and 4 labels (recall that there are 21 different values for the 4 attributes used in the test). As shown in this figure, 17 preference labels (more than 80%) are perfectly adapted (0 labels away) or are immediately next to the ideal label (only 1 label away), 4 preference values are 2 labels away, and no preference values are 3 or 4 labels away.

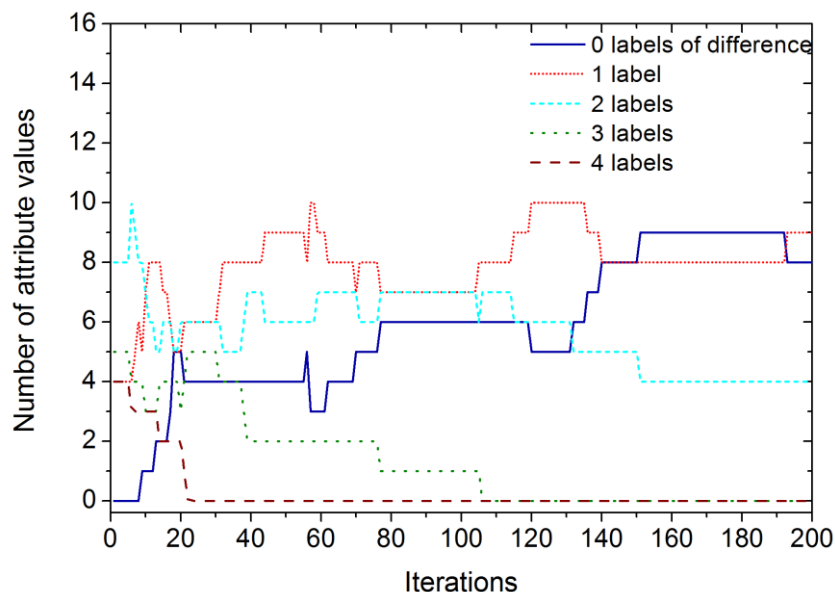


Figure 40. A quantitative study of the misclassified labels in the comparison of the current profile with the ideal profile

The proposed implicit adaptation algorithm introduces several parameters that should be properly customized. This part explains how these parameters influence the final result, although it should be noted that the exact parameter values may depend on the context in which the adaptation is taking place (e.g., number of attributes and number of values per attribute). The tests were performed using the following initial values for the parameters: 25% for the percentage of over-ranked alternatives (t), 3 for the number of changes of preferences at each iteration (pc), 5 for the number of pieces of evidence required to change a preference (k), 5 for the number of stored selected alternatives (h), and 5 for the number of minimum over ranked alternatives needed in the adaptation process (mo). The parameters were tested sequentially in the order provided in the following subsections. The best value for a particular parameter was used in the following tests.

Evaluation of the percentage of over-ranked alternatives (t)

Over-ranked characteristics are extracted by considering a threshold defined as a percentage of the number of over-ranked alternatives. Figure 41 shows that after performing tests using threshold values of 25%, 50%, and 75%, it was noticed that the threshold of 25% gave the best results.

Having a greater threshold means that the extracted characteristics need to have a greater repetition value; thus, fewer characteristics are extracted, which reduces the number of possible changes in the profile. On the contrary, the extraction of valuable information is compromised as the threshold decreases, because incorrect characteristics (which usually have a low number of repetitions) are extracted more easily.

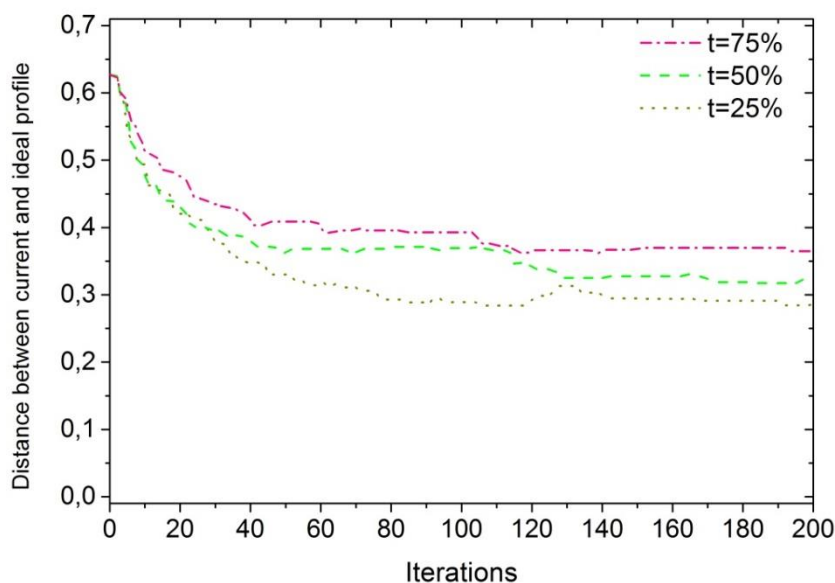


Figure 41. An empirical study of the influence of the characteristics extraction threshold

Evaluation of the number of preference changes at each iteration (pc)

In the previous adaptation mechanism part, a parameter that controls how many preference values can be increased/decreased during each adaptation iteration was introduced. Figure 42 compares the performance of the system with several values (1, 3 and 5). Setting the value to 3—which allowed the system to make up to three increases and three decreases for any preference value of the profile during each iteration—gave the best results. This number can depend on the domain in which the recommender framework is applied, but from this particular test, we can draw two conclusions. Using a low value means making the preference changes that have the greatest supporting evidence, but it also slows down the adaptation process. On the other hand, using a high value allows more adaptations to be made per adaptation step, which makes it easier for changes with a low amount of evidence to be made. Therefore, a compromise needs to be reached if an appropriate value is to be found for this parameter.

Evaluation of the evidence required to change a preference (k)

Previously, the parameter that indicates the amount of evidence required to increase/decrease a preference value in the profile was introduced. In this case, with low values, the amount of required evidence is small, and the quality of these changes is also compromised.

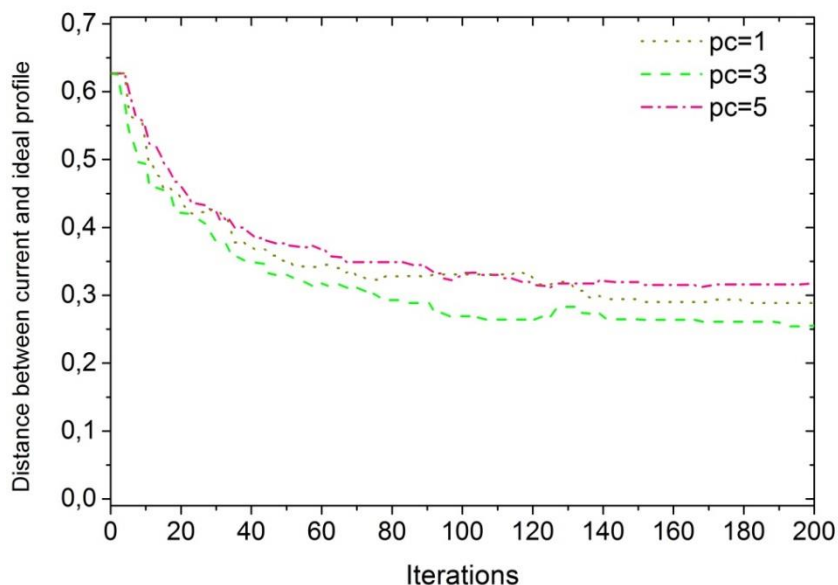


Figure 42. An empirical study of the influence of the maximum number of profile changes per iteration

On the other hand, higher values for this parameter make it harder to change the preferences, and more iterations are needed to reach a near-ideal profile. These differences, however, tend to decrease as the number of iterations increases. Figure 43 shows these behaviours and the good performance of an intermediate value such as ± 5 .

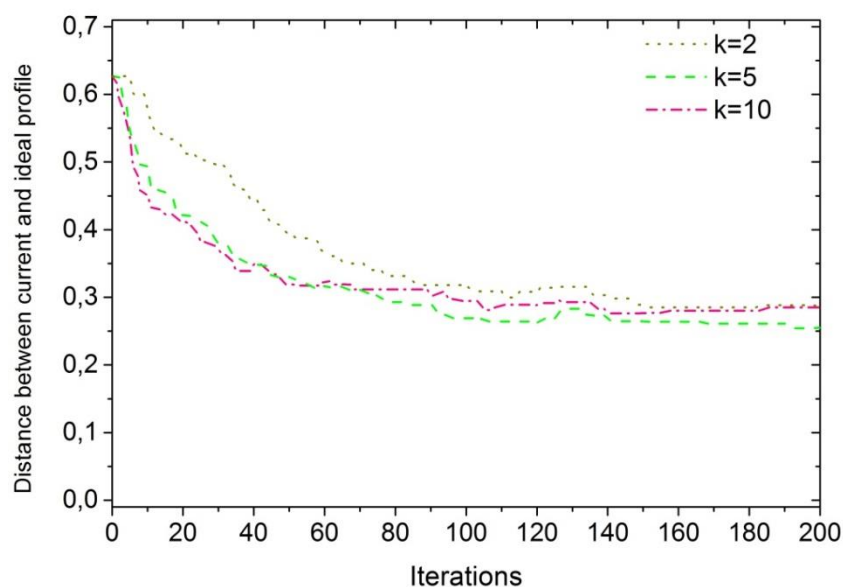


Figure 43. An empirical study of the influence of the level of evidence required to make a change

Evaluation of the number of stored selected alternatives (h)

As mentioned previously, the off-line process needs to consider a certain amount of information stored from previous recommendations if it is to discover possible user trends over time and modify the user preferences accordingly. Figure 44 shows the results obtained when this parameter is set to 2, 5 and 10. Storing 5 or 10 alternatives gives much better results than using only 2. However, it can be seen that the distance does not improve much between using 5 or 10; therefore, it can be concluded that storing the previous selections improves the performance of the algorithms up to a certain limit.

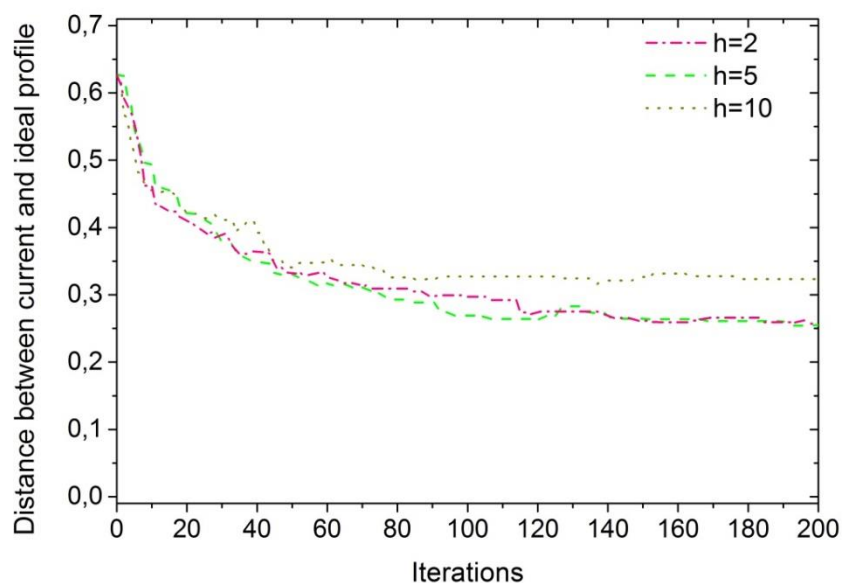


Figure 44. An empirical study of the influence of the minimum number of stored selections

Evaluation of the number of minimum over-ranked alternatives (mo)

The parameter mo determines the minimum number of over-ranked elements needed before the process of detecting common characteristics can start. Figure 45 represents the test results when a mo value of 2, 5 and 10 is used. The final result of the tests (iteration 200) shows that the greater the value of this parameter, the lower the distance to the ideal profile. This, however, can only be perceived after a certain number of iterations (50 in this case).

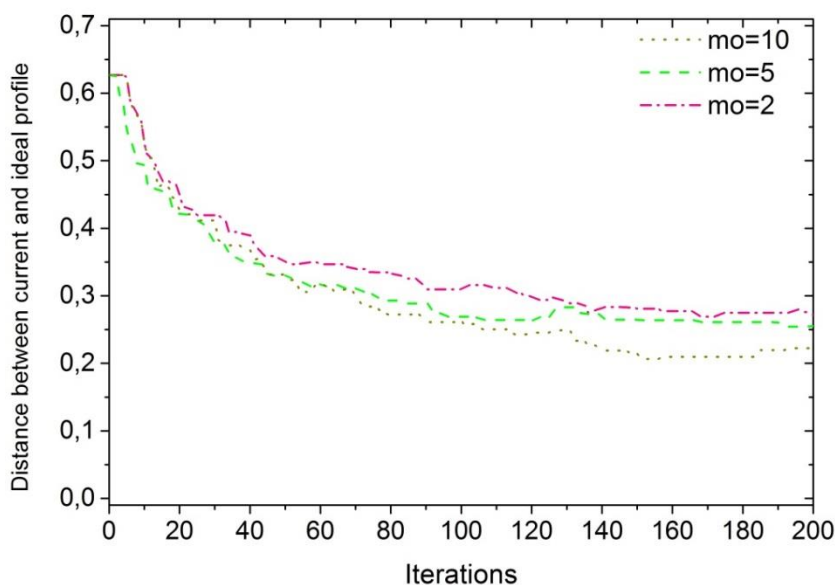


Figure 45. An empirical study of the influence of the minimum number of over-ranked alternatives

4.3.5 Preference learning on multi-valued attributes

The linguistic algorithm used to adapt categorical preferences explained previously needs some improvements to be able to manage *lists of values*. When single-valued attributes were considered, the user selection pointed directly towards the value the user liked for that attribute. Now, however, we cannot be sure which one/s of the values listed in the attribute is/are the one/s of interest for the user. That is the reason why it has been necessary to design a “*relevance function*” which indicates how relevant is a value found among the over ranked alternatives or in the selected alternative. The basic properties that the relevance function should satisfy are the following:

- The fewer values appear in a categorical attribute, the more relevant they are.
- A value present in the selected alternative has a relevance that is inversely proportional to the number of over ranked alternatives in which it appears. Moreover, the shorter the lists of categorical values in the over ranked alternatives are, the less relevant will be the value in the selected alternative.
- The relevance of a categorical value of an attribute will be stronger if the average number of values for that attribute is smaller than the average number of values for the other categorical attributes.

Relevance is measured in a $[0,1]$ scale, with 1 meaning maximum relevance. To calculate how relevant a term t of the attribute j is among the over ranked alternatives the following expression is used (the relevance value is 0 if it does not appear in any of the over ranked alternatives):

$$R_j^o(t) = \frac{1}{no} \sum_{i=1}^{nt} \frac{1}{nv_j^i} \quad (4.9)$$

Here, no represents the number of over ranked alternatives, nt the number of over ranked alternatives where t appears, and nv_j^i the number of values that appear for the attribute j in the alternative i . In this equation we consider that every linguistic term that appears in the over ranked alternatives has a relevance which is inversely proportional to the number of other values for the same attribute that appear among the over ranked alternatives that contain the term. Note that the maximum relevance among the over ranked alternatives is 1, in the case in which the term appears in all over ranked alternatives and it is the only value for the attribute in all of them. The minimum relevance is 0, when the term does not appear in the over ranked alternatives.

To calculate the relevance of a term in the selection the following formula is used (the relevance value is 0 if the term does not appear in the selection):

$$R_j^s(t) = \frac{1}{2} \left(\frac{1}{nv_j} + \frac{nl}{tv} \right) \quad (4.10)$$

Here nv_j represents the number of values that appear for the attribute j in the selection, nl the total number of linguistic attributes, and tv the total number of linguistic values that appear in the selection. The relevance of a term in the selection is the mean of the inverses of the number of values of the attribute in the selected alternative and the average number of values of all the categorical attributes in the selection. Note that, in fact, this formula does not depend on the term t ; therefore, all the terms in the selection have the same relevance. The maximum relevance is 1, if all the categorical attributes contain a single value in the selected alternative. The minimum relevance is 0, if the term does not appear in the selection.

Finally, after calculating both partial relevancies for all the terms, the overall relevance $R_j(t)$ is calculated as:

$$R_j(t) = R_j^s(t) - R_j^o(t) \quad (4.11)$$

Note that this overall relevance gives a value between -1 and 1. Negative values are associated to situations in which the term appears often in the over ranked alternatives. Positive values indicate that the term is important in the selection (it is probably in a short list of values) and does not appear very much in the over ranked alternatives.

With this final evaluation it can be decided if the preference value on a certain categorical value has to be increased or decreased. Considering a threshold γ to avoid making low-relevance changes in the profile, it can be deduced that:

- If $R_j(t) > \gamma$, the preference over the term t for the attribute j should be increased (moved to the next linguistic label).
- If $R_j(t) < -\gamma$, the preference over term t for the attribute j should be decreased (moved to the previous linguistic label).

The following example (see Figure 46) illustrates the computation of the relevance of different terms in a situation in which there are three over ranked alternatives (o_1 , o_2 and o_3) before the selection (s), each one formed by three categorical attributes (a_1 , a_2 and a_3).

<i>Ranked alternatives</i>			
	a_1	a_2	a_3
o_1	<i>A B C D</i>	<i>I J</i>	<i>M</i>
o_2	<i>A B E</i>	<i>G H I J K</i>	<i>N</i>
o_3	<i>A</i>	<i>G J K</i>	<i>L</i>
s	<i>A B C F</i>	<i>H</i>	<i>M</i>

Figure 46. Example of ranked alternatives

First, we will compare the relevance of terms *A*, *B*, *C* and *F* (values of attribute a_1 in the selection) among the over ranked alternatives, by applying Eq.(4.9):

$$R_j^o(A) = (1/3) * (1/4 + 1/3 + 1/1) = 0.528$$

$$R_j^o(B) = (1/3) * (1/4 + 1/3) = 0.194$$

$$R_j^o(C) = (1/3) * (1/4) = 0.083$$

$$R_j^o(F) = (1/3) * (0) = 0$$

It can be observed that terms *A* and *B* appear in the first two alternatives. However, in the third one *A* appears alone, increasing greatly its overall relevance. At the end, the relevance of *A* is more than twice the one of *B*. Then, it can be argued that it is likely that *A* is one the reasons that caused the user not to select any of the first three alternatives, so its level of preference may need to be readjusted negatively. Term *C* only appears in the first alternative and accompanied by many other terms, so its relevance on the over ranked set of alternatives is very low. Finally, term *F* does not appear in any over ranked alternative, so its relevance here is 0.

The next example consists in finding the overall relevance of a term. Let's evaluate the global relevance of the term *H* in the second attribute. To do this, it is necessary to calculate its relevance among the over ranked alternatives and in the selected alternative and then obtain the overall relevance using Eq.(4.11).

$$R_j^s(H) = (1/2) * ((1/1) + (3/6)) = 0.75$$

$$R_j^o(H) = (1/3) * (1/5) = 0.067$$

$$R(H) = 0.75 - 0.067 = 0.683$$

A positive overall relevance means that the term is well considered by the user (it may be the reason why the user selected the alternative). On the other hand, a negative relevance of a term indicates that it may have been the reason why the user did not choose the alternatives in which it appears. In this example, the term *H* has a very low relevance in the set of over ranked alternatives (it just appears once and in an alternative where that attribute has a high number of values). However, it appears alone in the selected alternative. This fact produces a very positive final relevance, meaning that *H* has a good chance to be the reason why the user selected

that alternative, so the preference learning system should readjust its preference positively.

4.4 Numeric and linguistic simultaneous learning

The learning processes of preferences over numeric and categorical preferences have no dependences between them and can be executed independently. That is, in the learning stage the task is divided in two parts: the first one involves the numeric preferences and the second one involves the categorical preferences (see Figure 47).

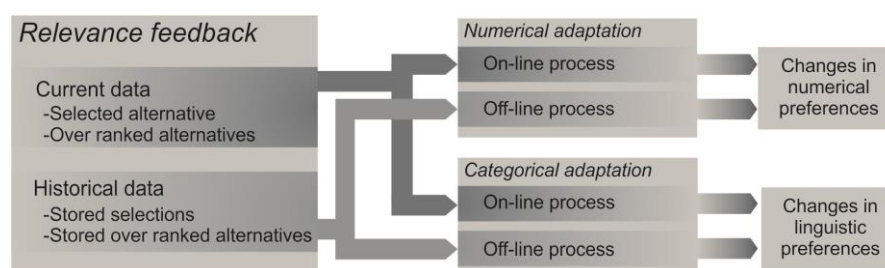


Figure 47. Independent execution of the adaptation processes

The extensive evaluation conducted in the next chapter shows the two adaptation processes working in the same system, including the learning of complex preference functions over numeric attributes and preferences over multi-valued categorical attributes.

4.5 Conclusions

In this chapter the contributions of this Thesis related to the processes involved in the dynamic learning of preferences over numeric and categorical attributes have been presented.

In the case of the numeric preference adaptation, a useful algorithm for learning the value of maximum preference of a numeric attribute, based on the idea of attraction and repulsion forces, has been presented. Then, the performance of the learning process, in its on-line and off-line modalities, has been evaluated with good results in a simulated touristic domain. Results have been shown in two ways: with the distance between the ideal profile and the profile that is being learned, and with the position of the user selection. In both cases, from the 25th iteration the results are very satisfactory. The evaluation also included tests in which the user interests changed over time and the algorithm was proven to be able to manage those changes successfully. Moreover, the learning process was later improved to allow learning a more complex numeric preference function (slope and delta values) rather than just the value of maximum preference.

Afterwards, the techniques for learning preferences over categorical attributes have been introduced, distinguishing again between the on-line and off-line processes. In this case, the learning processes have more parameters that tune the behaviour of the algorithm than in the numeric

part, so an extensive evaluation on those values was conducted to study their influence. The learning process on that kind of attributes is slower than for the numeric ones so they require a greater level of interaction with the user (about 90 iterations) to start giving good enough recommendations. However, it is still a low number of interactions if the recommender system is focused in daily activities.

The next chapter includes a real test scenario in which the two processes work at the same time to learn preferences in a multi-valued multi-attribute context.

Chapter 5

Case study:

Restaurant recommendation

In the previous chapter we introduced the learning techniques that, through the observation of the user selection when a recommendation is requested, discovered the user preferences and decided if it was necessary to make any change in the preferences stored in the user profile. The adaptation algorithms were defined, tested and evaluated independently depending on the type of attributes (numerical or categorical). After observing the adequate operation of the processes, two improvements were made: the learning of the five parameters involved in the complex numeric preference function (see section 4.2.4) and the learning of preferences on multi-valued categorical attributes (see section 4.3.5). In order to test the whole system with those two new improvements, data of the restaurants in Barcelona has been used to implement a RS with the ability to learn the users' interests from their selections.

In the first part of this chapter a description of the data is given. Then, a detailed explanation of the whole recommender and learning algorithm is given, as well as the preferences setup. Finally, the results of the evaluation are provided.

5.1 Barcelona restaurants data

The data used in this problem has been collected from the “BcnRestaurantes” web page². The data set contains pre-processed information about 3000 restaurants of Barcelona evaluated by 5 attributes: 3 categorical (“Type of food”- 15 values, “Atmosphere”- 13 values, “Special characteristics” – 12 values) and 2 numerical (“Average price”, “Distance to city centre”). Figure 48 shows the complete list of possible values for each categorical attribute and the times each term appears in the whole set of alternatives. For the numerical attributes, their domain and units are shown. Moreover, Figure 49 and Figure 50 show the distribution of the values for the two numerical attributes: Figure 49 displays a histogram of the distribution of the values of the “Distance to city center” attribute in intervals of 0.2 km from 0 to 10 km, and Figure 50 does the same with the attribute “Average price” but in the intervals “15 to 30 €”, “30 to 45 €”, “45 to 60 €” and “More than 60 €”. Some attributes and values from the original data were not considered due to their low relevance (number of appearances), and some of them were aggregated together to avoid considering similar values with different names. One

² Website: <http://www.bcnrestaurantes.com> (Last accessed: April 16th, 2013).

example of register in the data file is “Fonda España; {National, Season cuisine, Traditional}; {Classic, For families}; {Round tables, In a hotel, With views}; 45; 0.979”, being “Fonda España” the restaurant name, “National”, “Season cuisine” and “Traditional” the types of food served, “Classic” and “For families” the restaurant atmosphere, “Round tables”, “In a hotel” and “With views” other important restaurant characteristics, 45€ the average menu price, and 0.979 km the distance to the city centre.

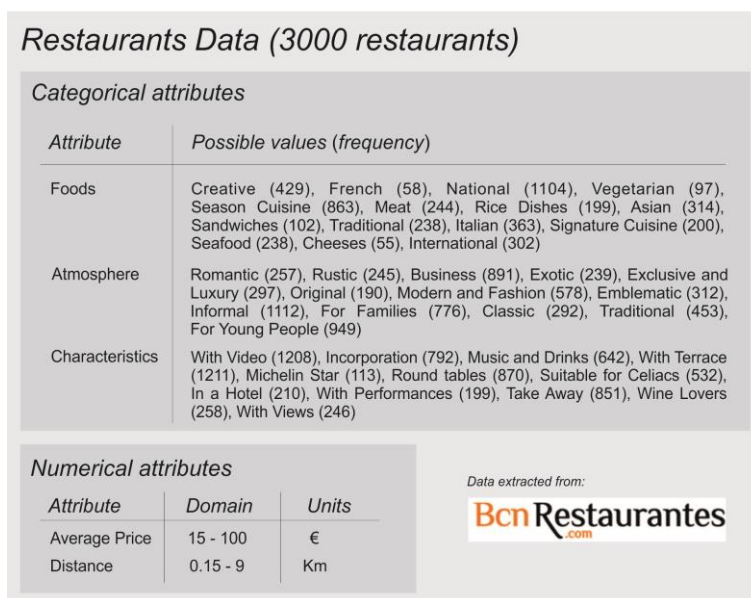


Figure 48. Restaurants data details

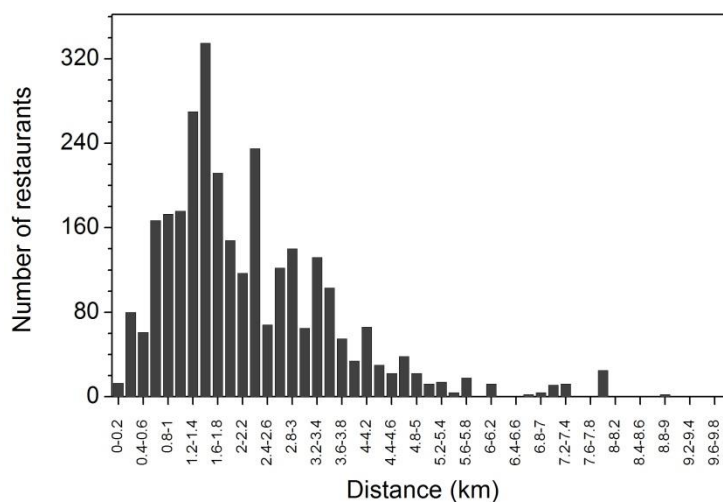


Figure 49. "Distance to city centre" values distribution

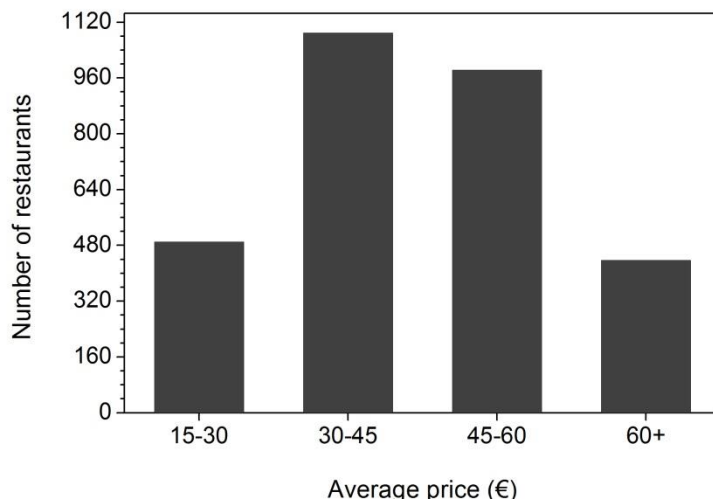


Figure 50. "Average price" values distribution

5.2 Recommendation and adaptation

The set of 3000 restaurants has been randomly divided in blocks of 15 alternatives that are ranked independently, which gives out a total of 200 different recommendations. An ideal profile was manually defined and three initial profiles were created randomly. The goal is to learn the ideal profile starting from these three different points. In this evaluation the preferences over the categorical attributes are represented with labels from a term set of 7 values, shown in Figure 51, which are "Very Low" (VL), "Low" (L), "Almost Low" (AL), "Medium" (M), "Almost High" (AH), "High" (H) and "Very High" (VH). The values of the ideal profile and the initial values of the three testing profiles are represented in Figure 52.

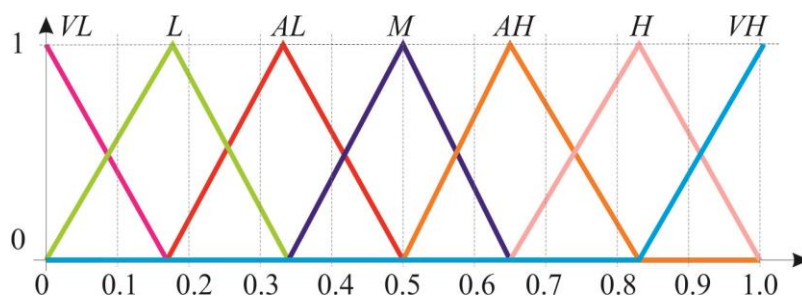


Figure 51. Linguistic preference label set of 7 values

The whole process (for each of the three profiles, repeated 200 times) consists in:

- 1) Ranking a set of 15 alternatives according to the current (initially random) profile.
- 2) Simulate the selection of the user by choosing the alternative that fits better with the ideal profile.
- 3) Extract relevance feedback from the selection (over ranked alternatives and the selection itself).
- 4) Decide which changes need to be made to the current profile and apply them.

Evaluation Profiles												
Preferences on numerical attributes												
Average price					Distance to city center							
	Ideal	P1	P2	P3		Ideal	P1	P2	P3			
v_{pref}	15	25	45	25	v_{pref}	0.3	0.2	5	7			
Δ_i	10%	20%	40%	30%	Δ_i	10%	20%	40%	30%			
Δ_r	10%	20%	40%	10%	Δ_r	10%	20%	40%	10%			
m_i	1	1	2	2	m_i	1	1	2	2			
m_r	1	1	2	0.5	m_r	1	1	2	0.5			

Preferences on categorical attributes														
Type of foods				Atmosphere				Special characteristics						
	Ideal	P1	P2	P3		Ideal	P1	P2	P3		Ideal	P1	P2	P3
Creative	M	VH	L	AL	Romantic	L	H	AL	AH	With video	VH	AL	VL	AH
French	VL	H	M	L	Rustic	AL	AL	M	M	Incorporation	M	VH	L	M
National	H	VH	VL	VH	Business	VL	VH	VL	H	With terrace	VH	VL	H	M
Vegetarian	VL	M	VL	L	Exotic	AH	VH	L	H	Michelin star	VL	VH	VL	L
Season cuisine	L	L	VL	H	Exclusive. Luxury	M	L	H	AL	Music and drinks	VH	M	M	AL
Meat	VH	VH	M	H	Original	AH	VL	L	VL	Round tables	M	AH	AH	H
Rice dishes	AH	AH	AH	L	Modern and fashion	H	H	AH	M	Suitable for celiacs	AL	VH	M	VL
Asian	M	VL	H	AL	Emblematic	AH	AH	VH	VH	In a hotel	L	L	AL	L
Sandwiches	VH	VL	AL	M	Informal	H	VH	VL	AL	With performances	AH	VH	VH	AL
Traditional	H	H	M	AL	For families	L	L	AL	H	Take away	M	VH	M	L
Italian	VH	VL	M	AL	Classic	L	H	H	L	Wine lovers	VL	VL	VH	AL
Signature cuisine	VL	H	AH	H	Traditional	M	M	M	L	With views	H	AL	AL	VL
Seafood	AL	AL	AH	VH	For young people	VH	L	L	L					
Cheeses	M	AH	L	VH										
International	H	L	AH	AL										

Figure 52. Three initial profiles and ideal profile used in the evaluation.

Some information about the whole process is stored after each iteration, including the position of the selected alternative, the distance between the ideal and current profiles, and the preferences over linguistic and numeric values. The execution time of the whole automatic evaluation process (with the adaptation of the three initial profiles) has been of 18 seconds with a computer equipped with an AMD Phenom 9550 Quad-Core processor (2.20 Ghz) and 4Gb of RAM. Provided that this process consists in 600 evaluations and ranking of alternatives (200 for each adapting profile) and a similar number of executions of the adaptation processes, it can be said that the time the user has to wait for a single recommendation or adaptation is unnoticeable (about 30ms) and does not compromise the user experience in real time with the platform.

5.3 Results evaluation

In order to evaluate the results of the new learning techniques, a distance function has been defined to calculate how different the profile we are learning is to an ideal profile which represents the exact preferences of the user. The first step is to calculate the distance for each attribute, taking into account if it is numeric or categorical. The distance between numeric attributes is calculated as:

$$d(n, P, I) = 1 - p_n^P(v_{pref_n}^I) \quad (5.1)$$

where n is the numerical attribute, P is the current profile (the one being learned), I is the ideal profile, and $p_n^P(v_{pref_n}^I)$ is the value of preference of the v_{pref} value for the attribute n in I using the preference function of the same attribute in the profile P . A distance 0 means that the v_{pref} values in both profiles are equal.

The equation to calculate the distance between categorical attributes is:

$$d(l, P, I) = \frac{1}{card(l)} \sum_{k=1}^{card(l)} \frac{|CoG(p_l^P(v_k)) - CoG(p_l^I(v_k))|}{|CoG(s_{min}) - CoG(s_{max})|} \quad (5.2)$$

where l is the categorical attribute, $card(l)$ is the cardinality of the attribute l (i.e., the number of different linguistic values it can take), $CoG(p_l^P(v_k))$ and $CoG(p_l^I(v_k))$ are the x -coordinate of the centres of gravity of the fuzzy linguistic labels associated to the value of preference of v_k in the profiles P and I , respectively, and $CoG(s_{min})$ and $CoG(s_{max})$ are the centers of gravity of the minimum and maximum labels of the domain, respectively. Finally, the distance between two profiles is calculated as:

$$D(P, I) = \frac{1}{na} \sum_{k=1}^{na} d(k, P, I) \quad (5.3)$$

where na is the total number of attributes.

During the three tests (one for each initial random profile) the distance between the adapting and the ideal profile has been calculated in each iteration. Figure 53 (continuous line) shows the average of the three distances. It can be seen that the initial average distance between the ideal and the adapting profiles is around 0.59. After 200 iterations it reaches a distance around 0.1. Although 200 iterations may seem a large number, it can also be observed that with only 50 iterations a very acceptable result of 0.2 is obtained.

To see to what extent the new approach to learn the complex numeric preference function explained in Section 4.2.4 (slope and delta values) has improved the result of the basic numeric adaptation algorithm, Figure 53 also compares the results with (continuous line) and without (dashed line) that functionality. It can be seen how the improvement has been noticeable (distance improvement of about 0.07).

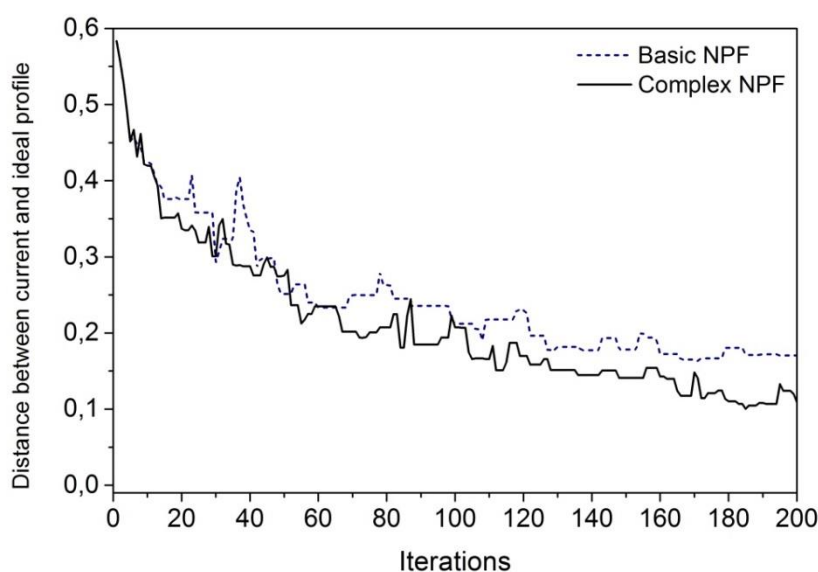


Figure 53. Average distance between the current and the ideal profile

Figure 54, Figure 55 and Figure 56 represent the learning evolution of three attribute values for the categorical attributes “Type of foods”, “Atmosphere” and “Special characteristics”, respectively, and show how the ideal values indicated in the ideal profile are learned through the iterations of the algorithm. This evolution, as the one of the numerical attributes shown later in this section, corresponds to the test with the initial profile P2 (see Figure 52).

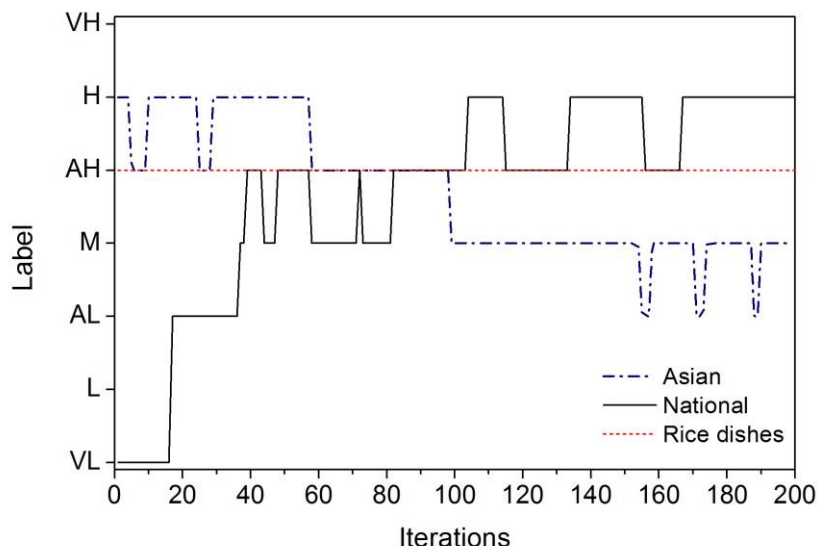


Figure 54. Learning evolution of the preference over three values of the attribute "Types of food"

Figure 54 shows how the preferences over the values “National”, “Asian” and “Rice dishes” of the attribute “Type of foods” are learned. “National” starts with a level of preference “Very Low” and stabilizes in the ideal value “High”(H). The value “Asian” starts with a “High” (H) level of preference and it is seen how it reaches the ideal value of “Medium” (M), although there are some changes between “Medium” (M) and “Almost Low” (AL). For the “Rice dishes” value, since the initial and the ideal value are the same, it can be seen how it does not change at all.

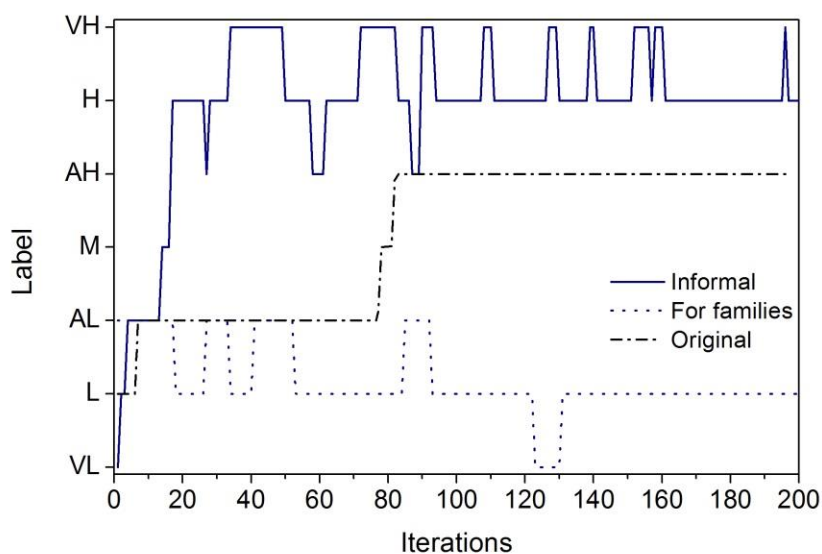


Figure 55. Learning evolution of preferences on the values of the attribute "Atmosphere"

Figure 55 shows the learning of the preferences on the values “Informal”, “For families” and “Original” of the attribute “Atmosphere”. “Informal” has initially a “Very Low” (VL) level of preference but the system quickly learns that the true preference over that parameter is “High” (H); however, there are several changes to a “Very High” (VH) preference. The “For families” value starts with an “Almost Low”(AL) preference, which is very similar to the ideal “Low”, so there are not many changes here. “Original” stabilizes in the correct level of preference (“Almost High” (AL)) in less than 90 iterations, starting from “Low” (L).

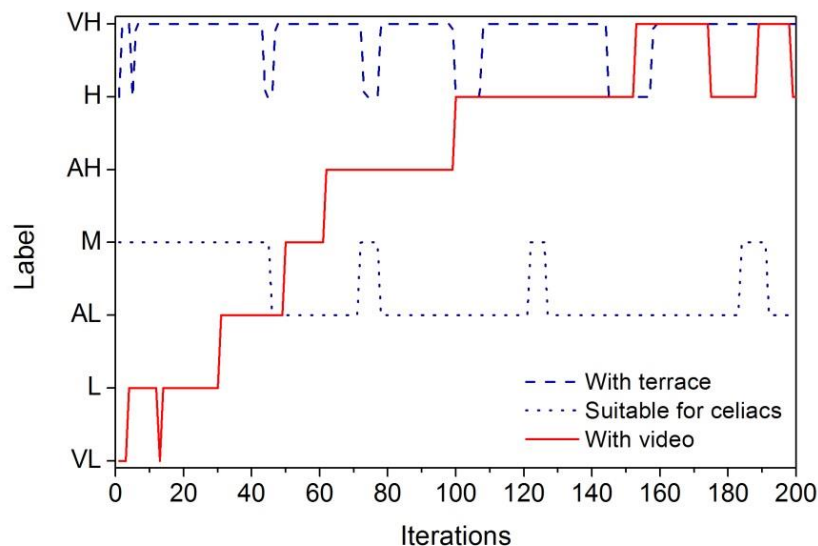


Figure 56. Learning evolution of the preferences on the values of the attribute "Special characteristics"

Figure 56 shows how the preferences over the values “With terrace”, “Suitable for celiacs” and “With video” of the attribute “Special characteristics” are learned. The value “With terrace” starts with a “High” (H) level of preference, which is very similar to the ideal “Very High”, and it is quickly learned without many changes. The value “Suitable for celiacs” has an initial preference of “Medium” (M) that reaches the ideal “Almost Low” (AL) in less than 50 iterations. Finally, the preference on the value “With video”, which initially has a value of “Very Low” (VL) that is the opposite of the ideal one (“Very High” (VH)), is correctly learnt through the evaluation process.

Figure 57 and Figure 58 represent the evolution of the preferred value for the numerical attributes “Average price” and “Distance to city center”, respectively, and show how the ideal value of preference of the numeric function indicated in the ideal profile is learned through the 200 iterations of the algorithm.

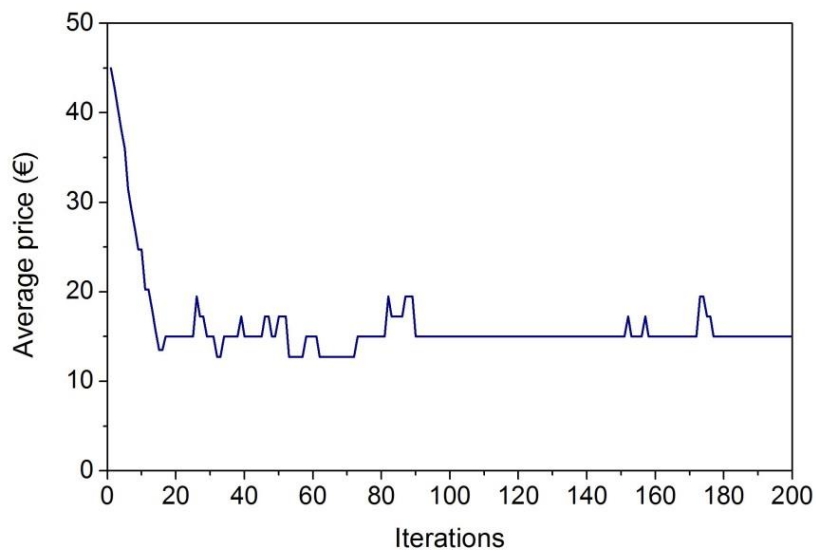


Figure 57. Evolution of the v_{pref} value of the attribute "Average price"

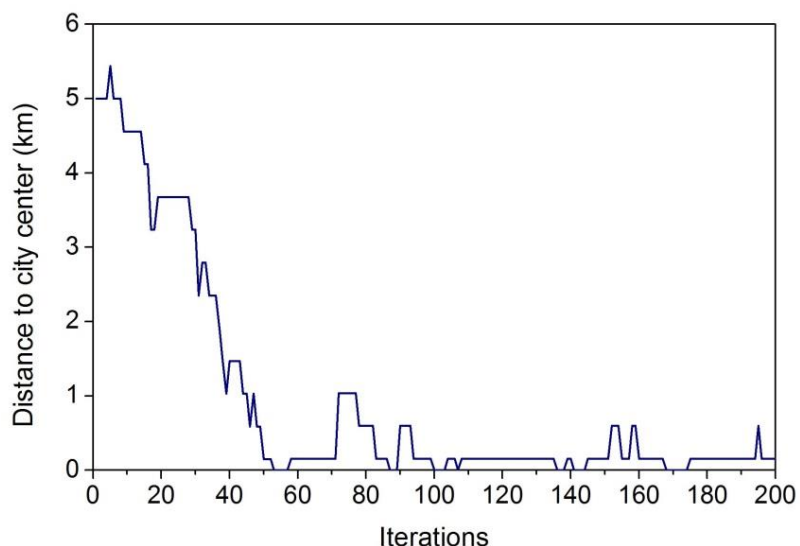


Figure 58. Evolution of the v_{pref} value of the attribute "Distance to city center"

In the figure that represents the evolution of the value of maximum preference of the attribute "Average price" (see Figure 57), it can be seen how the ideal one (15€) is correctly learned in less than 20 iterations and it remains stable from there. In the case of the value of maximum preference of the attribute "Distance to city centre" (see Figure 58), a value very close to the ideal one (0.3 Km) is learned in 50 iterations, and although it does not remain as static as in the previous example, the oscillations are not serious enough to compromise the recommendation process.

Figure 59 shows the visual evolution of the numeric preference function of the attribute "Distance to city centre". The graphical representation of the function is shown before the start of the learning process (iteration 0), and then after the iterations 20, 50 and 100. In all pictures, the current learned function is compared with a dotted line representing the ideal preference function. The graph in Figure 59a is defined by the values of the third initial profile shown in Figure 52. After 20 iterations (Figure 59b), the value of maximum preference has moved towards 3 Km, the slopes are

$m_l = 1.8$ and $m_r = 0.6$ and the widths are $\Delta_l = 0.3$ and $\Delta_r = 0.12$. After iteration 50 (Figure 59c), the v_{pref} value is 1 Km, the slopes are $m_l = 1.5$ and $m_r = 0.6$ and the widths are $\Delta_l = 0.2$ and $\Delta_r = 0.1$. Finally, in Figure 59d, it can be seen how after 100 iterations the numeric preference function learned ($v_{pref} = 0.3$, $m_l = 1.2$, $m_r = 0.8$, $\Delta_l = 0.15$, $\Delta_r = 0.09$) almost perfectly fits the ideal one. Although the learning of a very similar function to the ideal one required about 100 iterations, it is not necessary to obtain it to start giving adequate recommendations since, as other tests shows, very good results are obtained in less than half of that number of iterations.

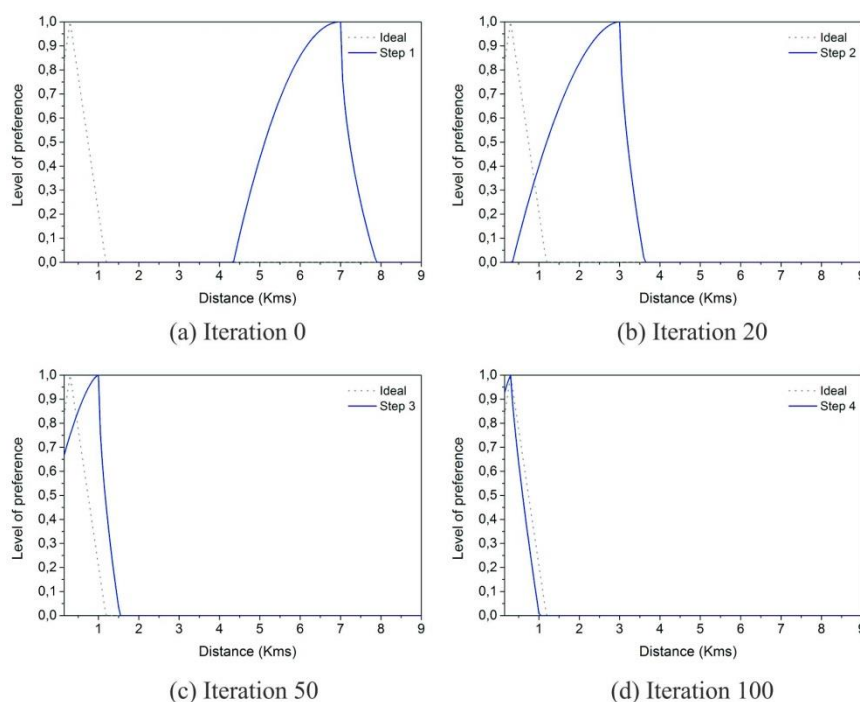


Figure 59. Evolution of “Distance to city centre” numeric preference function

To wrap up the results evaluation, Figure 60 shows in what position the user selection is being ranked by the RS on each of the iterations in the first test (the three give similar results). This figure shows the results in a more intuitive way. Notice that the system is accurate if the selected alternative is in the first positions of the 15-items list in each iteration. Many factors can interfere in the process and make the learning of the exact ideal profile a very hard task, but if the user selection appears in the first positions, it can be considered that the learning process is working properly. As it can be observed in Figure 60, after about 50 iterations, the selected alternative is among the first three ones in 95% of the cases (and the first one in around 70% of the cases).

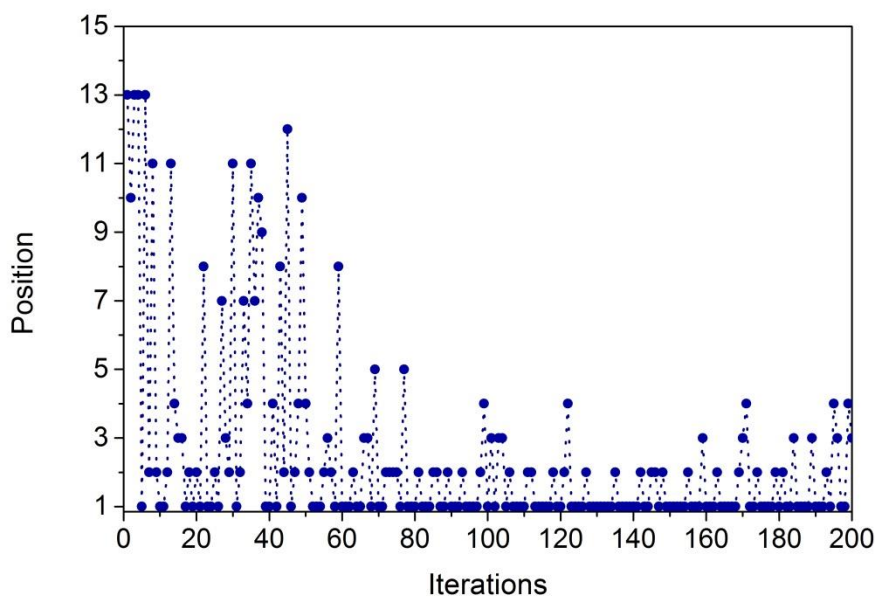


Figure 60. Position of the selected alternative in each iteration (Test 1)

5.4 Conclusions

In this chapter tests have been conducted to evaluate the performance of the learning algorithm in an environment where the alternatives are defined by numeric and categorical attributes.

Learning processes have proven to be accurate in that context showing good results in a relatively low number of interactions (iterations) with the user (about 60). It has also been shown how the numeric preference function is learnt adequately and, moreover, how the introduction of a complex function has improved the learning process. It has also been shown how linguistic preferences over categorical attributes are adequately learnt in a context where categorical attributes are multi-valued.

Chapter 6

Conclusions

The work developed in this doctoral thesis has shown that it is possible to efficiently learn in an unsupervised way the preferences of a user with regard to the values that can be taken by numerical or categorical attributes that describe a set of objects, in situations in which the user is constantly faced with a decision problem and the system can analyse the continuous selections of the user. The potential practical applications of the algorithms developed in this work are endless. For instance, a Web-based newspaper could analyse the news read by the user every day and learn quickly the kinds of news in which each user is interested, or a smart TV could analyse the programs watched by the user every evening to infer his/her preferred kind of entertainment. The ability to learn in a progressive way the preferences of the user permits the gradual improvement of the recommendations of the system, even in situations in which the user preferences may evolve on time.

The sections in this final chapter detail the specific contributions of the work, the publications derived from the work in the last years, and some open lines of future work.

6.1 Contributions

The work has made contributions in the following areas: Representation of Preferences, Aggregation Operators and Multi-Criteria Preference Learning in Recommender Systems.

The thesis started with the definition of a general and flexible framework that included all the components necessary to build personalised intelligent recommender systems: a user profile (containing the preferences of the user), a module capable of rating and ranking a set of alternatives according to these preferences, a module capable of showing the ranked list of options to the user and detecting his/her selection, and a final module smart enough to analyse the continuous selections of the user and update his/her preferences accordingly. This general framework allowed studying different ways of representing preferences, rating alternatives and updating the preferences.

Concerning the representation of the domain objects, the work has considered independently the management of numerical and (first uni-valued, later multi-valued) categorical attributes. Two preferential models on numerical attributes were considered: a simple initial one, in which the profile just keeps the preferred domain value for the user, and a final

complex one, in which a new kind of preferential functions based on five parameters was defined. In the case of categorical attributes, a linguistic preferential model in which a linguistic label is associated to each value was considered.

In order to rate the domain alternatives, it was decided to translate the numeric preferences (on numeric attributes) into linguistic labels, and merge the qualitative preferences on all the attribute values. In the area of aggregation two new operators have been designed: the Unbalanced Linguistic Ordered Weighted Averaging (ULOWA) operator and its extension to allow the induction of the aggregation order, the Induced ULOWA operator.

In this field all the existing linguistic operators use the order and the position of terms to aggregate the information. The proposed ULOWA operator provides a way of aggregating terms by considering the fuzzy membership functions that define them. This new operator is able to work with both balanced and unbalanced fuzzy sets. This fact gives the user more freedom when defining the sets according to his/her requirements. In this document it has been illustrated how this operator works and how it reacts to the change of one of the membership functions. Since it is based on the OWA operator, it permits to customize the aggregation results by using different policies, such as “at least half”, “as many as possible” or “average”.

The IULOWA operator is an extension of the IOWA operator which enables complex reordering processes to be carried out by using order-inducing variables in the context of unbalanced term sets. The variables which drive the ordering process are the specificity and the fuzziness of the fuzzy sets associated to linguistic labels. In the basis of that operator, a multi-person multi-criteria scenario has been presented, proposing a solution to a real decision making problem (environmental assessment in the Spanish research project SOSTAQUA) in two steps: 1) using the IULOWA to obtain a collective value for each criterion of each alternative; and 2) using the IULOWA to combine the aggregated values of the different criteria into a single overall evaluation.

After ranking the alternatives, showing them to the user and detecting his/her final selection, one of the basic contributions of the work has been the development of an efficient, autonomous, unsupervised, dynamic and domain-independent preference adaptation mechanism. The algorithm infers the reasons for selecting an alternative in front the others (and for discarding alternatives that were ranked above the selected one), and combines both present and past information in order to propose dynamic changes on the user’s interests.

Techniques for learning preferences over numeric attributes have been successfully designed and evaluated. A complex numeric preference function defined by 5 parameters (right and left slopes, right and left widths and value of maximum preference) has proven to be very adequate for expressing preferences about that kind of attributes. Moreover, the new learning algorithms are capable of shaping that function according to the user preferences.

The work has also contributed new preference adaptation techniques for the case of categorical attributes. In the initial steps of the thesis it was assumed that each object only had one value for each categorical attribute, but in a more advanced stage of the work the use and management of multi-valued categorical attributes has been successfully considered.

Both types of learning processes (categorical and numerical) have been tested and evaluated together in a real scenario (Barcelona restaurant recommendation) with promising results.

6.2 Publications

As indicated in the previous section, results obtained from the work conducted during this Thesis produced contributions in two research areas: *aggregation operators* and *multi-criteria preference learning* on recommender systems. During the elaboration of this Thesis the following main publications have been elaborated:

- 3 Accepted indexed Journal publications
 - *Information Sciences*
 - *Engineering Applications of Artificial Intelligence*
 - *Applied Intelligence*
- 2 Submitted indexed Journal publications
 - *Information Sciences*
 - *Knowledge Based Systems*
- 5 Congress publications
 - *IEEE World Congress on Computational Intelligence*
 - *European Society for Fuzzy Logic and Technology*
 - *International Conference on Agents and Artificial Intelligence*
 - *International Conferences of the Catalan Association for Artificial Intelligence*

In the area of aggregation operators two new operators (ULOWA and IULOWA) have been designed. Their respective definition and evaluation is included in the following publications:

- (Congress) Definition and evaluation of the ULOWA operator presented in the in the IEEE World Congress on Computational Intelligence in Barcelona, Spain, 2010 (ranking CORE “A”).

Isern, D., Marin, L., Valls, A., Moreno, A.: **The Unbalanced Linguistic Ordered Weighted Averaging Operator**. In: *IEEE World Congress on Computational Intelligence, WCCI 2010*, Barcelona, Catalonia, July 18-23 2010, pp. 3063-3070. IEEE Computer Society.

Abstract: Aggregation operators for linguistic variables usually assume a uniform and symmetrical distribution of the linguistic terms that define the variable. A well-known aggregation operator is the Linguistic Ordered Weighted Average (LOWA), which has been extensively applied. However, there are some problems where an unbalanced set of linguistic terms is more appropriate to describe the objects. In this paper we define the Unbalanced Linguistic Ordered Weighted Average (ULOWA) on the basis of the LOWA operator. ULOWA takes into account the fuzzy membership functions of the terms during the aggregation process. There is no restriction on the form of the membership functions of the terms, which can be triangular or trapezoidal, non symmetrical and non equally distributed. The paper demonstrates the properties of ULOWA. Finally, a comparison of this operator with some other aggregation operators for unbalanced sets of terms is done.

- (Congress) Definition and evaluation of the IULOWA operator presented in the 7th conference of the European Society for Fuzzy Logic and Technology (EUSFLAT) in Aix-Les-Bains, France, 2011.

Marin, L., Merigó, J.M., Valls, A., Moreno, A., Isern, D.: **Induced Unbalanced Linguistic Ordered Weighted Average**. In: Galichet, S., Montero, J., Mauris, G. (eds.) *7th conference of the European Society for Fuzzy Logic and Technology (EUSFLAT-2011)* and *LFA-2011*, Aix-les-Bains, France 2011, pp. 1-8. Atlantis Press.

Abstract: Aggregation operators for linguistic variables usually assume a uniform and symmetrical distribution of the linguistic terms that define the variable. This paper defines the Induced Unbalanced Linguistic Ordered Weighted Average (IULOWA). This aggregator takes into account the fuzzy membership functions of the terms during the aggregation operations of the pairs of terms. There is no restriction on the form of the membership functions of the terms, which can be triangular or trapezoidal, non-symmetrical and non-equally distributed. Moreover, the paper proposes to use the specificity and fuzziness measures of the terms to induce the order of the arguments, providing some examples of this criterion in decision making.

- (Journal) Extension of the definition of the IULOWA operator including the multi-person IULOWA definition as well as an extensive case study. Submitted to the *Information Sciences Journal* for revision (I.F.: 2.833).

Marin, L., Valls, A., Isern, D., Moreno, A.: **Induced Unbalanced Linguistic Ordered Weighted Average and its Application in Multi-Person Decision Making**. Submitted to *Information Sciences*.

Abstract: Linguistic variables are very useful for evaluating alternatives in decision making because they provide a vocabulary in natural language rather than numbers. Some aggregation operators for linguistic variables force the use of a symmetric and uniformly distributed set of terms. The need to relax these conditions has recently been posited. This paper presents the Induced Unbalanced Linguistic Ordered Weighted Average (IULOWA) operator. This operator can deal with a set of unbalanced linguistic terms that are represented using fuzzy sets (with a non-symmetric and non-uniform distribution). We propose a new order-inducing criterion based on the specificity and fuzziness of the different linguistic terms. Different relevancies are given to the fuzzy values according to their uncertainty degree. To illustrate the behaviour of the precision-based IULOWA operator, we present an environmental assessment case study in which a multi-person multi-criteria decision making model is applied.

In the area of preference learning the main publications are as follows:

- (Journal) Preference learning techniques considering single-valued categorical attributes evaluated in a real case scenario of recommending news articles of *The New York Times*, published in the *Information Sciences* journal (I.F.: 2.833)

Marin, L., Isern, D., Moreno, A., Valls, A.: **On-line dynamic adaptation of fuzzy preferences**. *Information Sciences*. 220, 5-21 (2013). doi:10.1016/j.ins.2011.10.008

Abstract: Recommender systems are very useful in domains in which a large amount of continuous information needs to be evaluated before a decision is made. Systems that permanently interact with users need to be adapted to changes in their interests. This paper proposes an algorithm that takes advantage of the preference information implicit in the actions of the user to dynamically adapt the user profile, in which user preferences are represented as fuzzy sets. The algorithm has been tested with real data extracted from the New York Times and has shown promising results. This paper presents the adaptation algorithm and discusses the influence of its basic parameters

- (Journal) Preference learning techniques on numerical attributes evaluated in a case scenario of recommending touristic destinations, accepted for publication in the *Applied Intelligence* journal (I.F.: 0.849)

Marin, L., Isern, D., Moreno, A.: **Dynamic adaptation of numerical attributes in a user profile**. *Applied Intelligence*. (2013). doi:10.1007/s10489-012-0421-5. *In Press*.

Abstract: Recommender systems try to help users in their decisions by analyzing and ranking the available alternatives according to their preferences and interests, modeled in user profiles. The discovery and dynamic update of the users' preferences are key issues in the development of these systems. In this work we propose to use the information provided by a user during his/her interaction with a recommender system to infer his/her preferences over the criteria used to define the decision alternatives. More specifically, this paper pays special attention on how to learn the user's preferred value in the case of numerical attributes. A methodology to adapt the user profile in a dynamic and automatic way is presented. The adaptations in the profile are performed after each interaction of the user with the system and/or after the system has gathered enough information from several user selections. We have developed a framework for the automatic evaluation of the performance of the adaptation algorithm that permits to analyze the influence of different parameters. The obtained results show that the adaptation algorithm is able to learn a very accurate model of the user preferences after a certain amount of interactions with him/her, even if the preferences change dynamically over time.

- (Journal) The whole recommendation framework with the learning of complex numerical preference functions and multi-valued categorical preferences, tested in a real case scenario (Barcelona restaurants recommendation) has been submitted to the *Knowledge Based Systems* journal (I.F.: 2.422).

Marin, L., Moreno, A., Isern, D.: **Automatic preference learning on numeric and multi-valued categorical attributes**. Submitted to *Knowledge Based Systems*.

Abstract: One of the most challenging tasks in the development of recommender systems is the design of techniques that can infer the preferences of users through the observation of their actions. Those preferences are essential to obtain a satisfactory accuracy in the recommendations. Preference learning is especially difficult when attributes of different kinds (numeric or linguistic) intervene in the problem, and even more when they take multiple possible values. This paper presents an approach to learn user preferences over numeric and multi-valued linguistic attributes through the analysis of the user selections. The learning algorithm has been tested with real data on restaurants, showing a very good performance.

- (Congress) Related previous studies on preference learning were presented in the *3rd International Conference on Agents and Artificial Intelligence (ICAART 2011)* and in the *13th and 14th International Conferences of the Catalan Association for Artificial Intelligence (CCIA 2010 and 2011)*.

Marin, L., Isern, D., Moreno, A.: **Unsupervised adaptation of the user interests**. In: *3rd International Conference on Agents and Artificial Intelligence*, Rome, Italy 2011, pp. 337-342. INSTICC Press

Abstract: One of the main problems in recommender systems is to ensure the quality of the user profile. This issue is particularly challenging if the user preferences may vary in time. This paper proposes a novel unsupervised algorithm to adapt dynamically the user profile, taking into account the interaction of the user with the system. The paper discusses the influence of the basic parameters of the adaptation algorithm and presents some promising preliminary results.

Marin, L., Moreno, A., Isern, D.: **Automatic learning of preferences in numeric criteria**. In: *14th International Conference of the Catalan Association for Artificial Intelligence, CCIA 2011*, Lleida 2011, pp. 120-129. IOS Press.

Abstract: Due to the astonishing speed at which new content is created and published on the Web, it is increasingly difficult for users to make the most appropriate decisions in front of an overwhelming amount of information. Recommender systems try to help users by analyzing and ranking the available alternatives according to their preferences and interests, modeled in user profiles. One important problem to solve in the development of these systems is how to discover the user preferences, and how to maintain them dynamically. In this work we propose to use the information given by a user in his/her interaction with the recommender system (e.g. the selection of the news to be read every morning) to infer his/her preferences on several criteria on which the decision alternatives are defined. More

specifically, the paper is focused in learning the most preferred value for the user in the case of numerical attributes. A methodology to adapt the user profile in a dynamic and automatic way is presented. The adaptations may be performed after each interaction of the user or after the system has gathered enough information from several user selections. We have developed a framework for the automatic evaluation of the performance of the adaptation algorithm that permits to analyze the influence of different parameters. The obtained results show that the adaptation algorithm is able to learn a very accurate model of the user preferences after a certain amount of interactions.

Marin, L., Isern, D., Moreno, A.: **A Generic User Profile Adaptation Framework**. In: *13th International Conference of the Catalan Association for Artificial Intelligence, CCIA 2010*, l'Espluga de Francolí, Tarragona, Spain 2010. Frontiers in Artificial Intelligence and Applications, pp. 143-152. IOS Press.

Abstract: The paper presents a recommender system that permits to manage user preferences using linguistic criteria and, after collecting information about selections made by the user, it performs an unsupervised adaptation of the user profile. It has been implemented as a Web application and designed in a generic way so that it can be applied to any decision making problem. It includes two separate modules: a module to rate and rank all alternatives received by the system according to the current interests of the user, and a module to adapt the current user profile in an unsupervised fashion collecting implicit information about the user interaction with the system. The paper presents some preliminary results and discusses the performance of the adaptation algorithm.

Finally, during the elaboration of this Thesis, as a result of a collaboration with a related project in the area of preference learning on recommender systems, the following article has been published:

- (Journal) Collaboration in the SigTur/E-Destination ontology-based touristic recommender, published in the *Engineering Applications of Artificial Intelligence* journal (I.F.: 1.665).

Moreno, A., Valls, A., Isern, D., Marin, L., Borràs, J.: **SigTur/E-Destination: Ontology-based personalized recommendation of Tourism and Leisure Activities**. *Engineering Applications of Artificial Intelligence*. 26(1), 633–651 (2013). doi:10.1016/j.engappai.2012.02.014.

Abstract: SigTur/E-Destination is a Web-based system that provides personalized recommendations of touristic activities in the region of Tarragona. The activities are properly classified and labeled according to a specific ontology, which guides the reasoning process. The recommender takes into account many different kinds of data: demographic information, travel motivations, the actions of the user on the system, the ratings provided by the user, the opinions of users with similar demographic characteristics or similar tastes, etc. The system has been fully designed and implemented in the Science and Technology Park of Tourism and Leisure. The paper presents a numerical evaluation of the correlation between the recommendations and the user's motivations, and a qualitative evaluation performed by end users.

6.3 Future work

Some future lines of research can be devised in the areas studied in this Thesis.

In the area of aggregation operators, a further study on the performance of the aggregation policies and their influence in the recommendation process can be made. Research in this area could result at finding if it is possible to include the **aggregation policies** as personal parameters in the user profile so, depending on the user who is asking for the recommendation, the evaluation of the alternatives will be done differently.

In the area of preference learning, some extensions can be done in the areas studied in this work.

First, the **initial profile** has been considered to be empty or generated randomly, which leaves all the responsibility to the learning process. Some techniques for generating good enough initial profiles can be studied, such as profiling new users by analysing his/her demographic properties or studying their similarity with other users that already have an initialised profile.

Another interesting question to study is the consideration of new kinds of attributes. In particular, **semantic attributes** (the ones that could take values inside an ontology), through which we could define mechanisms to allow the refinement of user preferences in that kind of attributes. For example, in a news recommendation system, if the learning algorithm finds out that the user likes “Sports”, refinement techniques could then deduce in which concrete sports the user is interested in and/or in what football teams the user is interested in if he likes this sport. Work in this area will be framed in the new Spanish research project called SHADE³ (Semantics and Hierarchical Attributes in Decision Making), which main objective is the development of new techniques to solve some of the current limitations of decision support systems. In SHADE, the attributes that define the objects of the considered domains are structured in a hierarchy and are not independent among them such as the ones considered in this work.

It is also interesting the study of preference learning in situations where the objects are just defined in a **textual** way rather than through attributes. Some research has been done in the Master Thesis of David Perelló. This work, directed by Dr. Antonio Moreno and Lucas Marin, has obtained positive preliminary results and is the first step at studying situations in which we do not have numerical, categorical neither semantic attributes.

Finally, another important future work is the implementation of the developed learning techniques in a **real environment** to assist real users in quotidian decision problems. This action could provide useful feedback that could help in the improvement of the algorithms and reveal new challenges in that area.

³ SHADE webpage: <http://deim.urv.cat/~itaka/SHADE> (Last accessed: May 15th, 2013)

References

- Adomavicius, G., YoungOk, K.: New Recommendation Techniques for Multicriteria Rating Systems. *IEEE Intell. Syst.* **22**(3), 48-55 (2007)
- Arias, J.J.P., Fernández Vilas, A., Díaz Redondo, R.P., Gil Solla, A., Ramos Cabrer, M., García Duque, J.: Making the most of TV on the move: My newschannel. *Inf. Sci.* **181**(4), 855-868 (2011). doi:10.1016/j.ins.2010.10.017
- Baltrunas, L., Amatriain, X.: Towards Time-Dependant Recommendation based on Implicit Feedback. In: Adomavicius, G., Ricci, F. (eds.) *Workshop on Context-aware Recommender Systems in conjunction with the 3rd ACM Conference on Recommender Systems, RecSys 09, New York, USA 2009*, pp. 25-30
- Basiri, J., Shakery, A., Moshiri, B., Zi Hayat, M.: Alleviating the cold-start problem of recommender systems using a new hybrid approach. In: 2010, pp. 962-967
- Basu, C., Hirsh, H., Cohen, W.: Recommendation as classification: Using social and content-based information in recommendation. In: 1998, pp. 714-720
- Batet, M., Moreno, A., Sánchez, D., Isern, D., Valls, A.: Turist@: agent-based personalised recommendation of touristic activities. *Expert Syst. Appl.* **39**(8), 7319-7329 (2012). doi:10.1016/j.eswa.2012.01.086
- Beliakov, G., James, S.: Induced ordered weighted averaging operators. In: Yager, R.R., Kacprzyk, J., Beliakov, G. (eds.) *Recent Developments in the Ordered Weighted Averaging Operators: Theory and Practice*, vol. 265. *Recent Developments in the Ordered Weighted Averaging Operators: Theory and Practice*, pp. 29-47. Springer Berlin, Heidelberg (2011)
- Beliakov, G., Pradera, A., Calvo, T.: *Aggregation Functions: A Guide for Practitioners*. Springer-Verlag, Berlin (2007)
- Blanco-Fernández, Y., López-Nores, M., Pazos-Arias, J.J., García-Duque, J.: An improvement for semantics-based recommender systems grounded on attaching temporal information to ontologies and user profiles. *Eng. Appl. Artif. Intell.* **24**(8), 1385-1397 (2011). doi:10.1016/j.engappai.2011.02.020
- Bonissone, P., Decker, K.: Selecting uncertainty calculi and granularity: An experiment in trading-off precision and complexity. In: Kanal, L.N., Lemmer, J.F. (eds.) *First Annual Conference on Uncertainty in Artificial Intelligence, UAI-85, New York, NY 1985*, pp. 217-248. Elsevier Science
- Boone, G.: Concept features in Re:Agent, an intelligent Email agent. In: Sycara, K.P., Wooldridge, M. (eds.) *2nd International Conference on Autonomous Agents and Multi Agent Systems, AGENTS 98, Minneapolis, Minnesota, US 1998*, pp. 141-148. ACM Press

- Borràs, J., Valls, A., Moreno, A., Isern, D.: Ontology-based management of uncertain preferences in user profiles. In: Pirlot, M., Mousseau, V. (eds.) DA2PL'2012 from Multiple Criteria Decision Aid to Preference Learning, Mons, Belgium, November 15-16 2012, pp. 83-89
- Burke, R.: Hybrid Web Recommender Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web, vol. 4321. Lecture Notes in Computer Science, pp. 377-408. Springer Berlin / Heidelberg, (2007)
- Cabrerizo, F.J., Alonso, S., Herrera-Viedma, E.: A consensus model for group decision making problems with unbalanced fuzzy linguistic information. *Int. J. Inf. Tech. Dec. Making* **8**(1), 109-131 (2009)
- Cabrerizo, F.J., Pérez, I.J., Herrera-Viedma, E.: Managing the consensus in group decision making in an unbalanced fuzzy linguistic context with incomplete information. *Know.-Based Syst.* **23**(2), 169-181 (2010)
- Castellano, G., Castiello, C., Dell'Agnello, D., Fanelli, A.M., Mencar, C., Torsello, M.A.: Learning Fuzzy User Profiles for Resource Recommendation. *Int. J. Unc. Fuzz. Know.-Based Syst.* **18**(4), 389-410 (2010). doi:10.1142/S0218488510006611
- Codina, V., Ceccaroni, L.: Taking Advantage of Semantics in Recommendation Systems. In: Alquezar, R., Moreno, A., Aguilar, J. (eds.) 13th International Conference of the Catalan Association for Artificial Intelligence, CCIA 2010, Espluga de Francolí, Catalunya 2010. *Frontiers in Artificial Intelligence and Applications*, pp. 163-172. IOS Press
- Chen, L., Sycara, K.: WebMate: A personal agent for browsing and searching. In: 1998, pp. 132-139
- Chen, S.-J.: A New Similarity Measure for Generalized Fuzzy Numbers Based on Geometric-Mean Averaging Operator. In: 16th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2006, Vancouver, CA 2006, pp. 1879-1886. IEEE Press
- Chen, S.J., Chen, S.M.: Fuzzy risk analysis based on similarity measures for generalized fuzzy numbers. *IEEE Trans. Fuzzy Syst.* **11**(1), 45-56 (2003). doi:10.1109/TFUZZ.2002.806316
- Chen, S.M.: New methods for subjective mental workload assessment and fuzzy risk analysis. *Cybernetics and systems, an International Journal* **27**(5), 449-472 (1996)
- Cheng, E., Jing, F., Li, M., Ma, W., Jin, H.: Using Implicit Relevance Feedback to Advance Web Image Search. In: IEEE International Conference on Multimedia and Expo, ICME 2006, Toronto, ON, Canada 2006, pp. 1773-1776. IEEE Computer Society
- Chiclana, F., Herrera-Viedma, E., Herrera, F., Alonso, S.: Some induced ordered weighted averaging operators and their use for solving group decision-making problems based on fuzzy preference relations. *Eur. J. Oper. Res.* **182**(1), 383-399 (2007)
- De Luca, A., Termini, S.: A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Inf. Control* **20**(4), 301-312 (1972). doi:Doi: 10.1016/s0019-9958(72)90199-4
- Delgado, M., Verdegay, J.L., Vila, M.A.: On aggregation operations of linguistic labels. *Int. J. Intell. Syst.* **8**(3), 351-370 (1993)
- Eyharabide, V., Amandi, A.: Ontology-based user profile learning. *Appl. Intell.* **36**(4), 857-869 (2012). doi:10.1007/s10489-011-0301-4
- Fan, W., Gordon, M.D., Pathak, P.: Effective profiling of consumer information retrieval needs: a unified framework and empirical

- comparison. *Dec. Supp. Syst.* **40**(2), 213-233 (2005). doi:10.1016/j.dss.2004.02.003
- Fürnkranz, J., Hüllermeier, E.: Pairwise Preference Learning and Ranking. In: Lavrac, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) 14th European Conference on Machine Learning, Cavtat, Croatia 2003, pp. 145-156. Springer Verlag
- Gantner, Z., Freudenthaler, C., Rendle, S., Schmidt-Thieme, L.: Optimal Ranking for Video Recommendation. In: Daras, P., Ibarra, O.M. (eds.) 1st International ICST Conference on User Centric Media, UCMedia 2009, Venice, Italy 2009. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp. 255-258. Springer
- Garcia Esparza, S., O'Mahony, M.P., Smyth, B.: Mining the real-time web: A novel approach to product recommendation. *Know.-Based Syst.* **29**, 3-11 (2012). doi:10.1016/j.knosys.2011.07.007
- Garibaldi, J.M., Ifeachor, E.C.: The Development of a Fuzzy Expert System for the Analysis of Umbilical Cord Blood. In: Szczepaniak, P., Lisboa, P.J.G., Kacprzyk, J. (eds.) *Fuzzy Systems in Medicine*, vol. 41. Studies in Fuzziness and Soft Computing, pp. 652-668. Physica-Verlag Heidelberg, (2000)
- Garmendia, L., Yager, R.R., Trillas, E., Salvador, A.: A t-norm based specificity for fuzzy sets on compact domains. *International Journal of General Systems* **35**(6), 687-698 (2006). doi:10.1080/03081070600867054
- Herrera-Viedma, E., Cabrerizo, F., Pérez, I., Cobo, M., Alonso, S., Herrera, F.: Applying Linguistic OWA Operators in Consensus Models under Unbalanced Linguistic Information. In: Yager, R., Kacprzyk, J., Beliakov, G. (eds.) *Recent Developments in the Ordered Weighted Averaging Operators: Theory and Practice*, vol. 265. Studies in Fuzziness and Soft Computing, pp. 167-186. Springer Berlin / Heidelberg, (2011)
- Herrera, F., Herrera-Viedma, E.: Linguistic decision analysis: Steps for solving decision problems under linguistic information. *Fuzzy Sets Syst.* **115**(1), 67-82 (2000)
- Herrera, F., Herrera-Viedma, E., Martínez, L.: A Fuzzy Linguistic Methodology to Deal With Unbalanced Linguistic Term Sets. *IEEE Trans. Fuzzy Syst.* **16**(2), 354-370 (2008)
- Herrera, F., Herrera-Viedma, E., Verdegay, J.L.: Direct approach processes in group decision making using linguistic OWA operators. *Fuzzy Sets Syst.* **79**(2), 175-190 (1996)
- Herrera, F., Martínez, L.: A 2-tuple fuzzy linguistic representation model for computing with words. *IEEE Trans. Fuzzy Syst.* **8**(6), 746-752 (2000)
- Herrera, F., Martínez, L.: A model based on linguistic 2-tuples for dealing with multigranular hierarchical linguistic contexts in multi-expert decision-making. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **31**(2), 227-234 (2001)
- Hong, T.-P., Chen, J.-B.: Finding relevant attributes and membership functions. *Fuzzy Sets Syst.* **103**(3), 389-404 (1999)
- Hsieh, C.H., Chen, S.H.: Similarity of generalized fuzzy numbers with graded mean integration representation. In: 8th International Fuzzy Systems Association World Congress, IFSA 99, Taipei, Taiwan, Republic of China 1999, pp. 551-555
- Isern, D., Marin, L., Valls, A., Moreno, A.: The Unbalanced Linguistic Ordered Weighted Averaging Operator. In: *IEEE World Congress*

- on Computational Intelligence, WCCI 2010, Barcelona, Catalonia, July 18-23 2010, pp. 3063-3070. IEEE Computer Society
- Jannach, D.: Finding Preferred Query Relaxations in Content-based Recommenders. In: Intelligent Systems, 2006 3rd International IEEE Conference on, Sept. 2006 2006, pp. 355-360
- Jensen, F.V.: An Introduction to Bayesian Networks. In. New York, (1996)
- Joachims, T., Radlinski, F.: Search Engines that Learn from Implicit Feedback. *Computer* **40**(8), 34-40 (2007)
- Kaya, T., Kahraman, C.: An integrated fuzzy AHP-ELECTRE methodology for environmental impact assessment. *Expert Syst. Appl.* **38**(7), 8553-8562 (2011). doi:DOI: 10.1016/j.eswa.2011.01.057
- Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum* **37**(2), 18-28 (2003). doi:10.1145/959258.959260
- Kim, J.S., Sohn, B.A., Whang, B.G.: A tolerance approach for unbalanced economic development policy-making in a fuzzy environment. *Inf. Sci.* **148**(1-4), 71-86 (2002)
- Klir, G.J.: Developments in Uncertainty-Based Information. In: Marshall, C.Y. (ed.) *Advances in Computers*, vol. Volume 36. pp. 255-332. Elsevier, (1993)
- Klir, G.J., Yuan, B.: On nonspecificity of fuzzy sets with continuous membership functions. In: 15th International Conf. on Systems, Man, and Cybernetics, Vancouver, Canada 1995, pp. 25-29. IEEE Computer Society
- Krulwich, B.: Lifestyle finder: Intelligent user profiling using large-scale demographic data. *AI Mag.* **18**(2), 37-45 (1997)
- Krulwich, B., Burkey, C.: Learning user information interests through extraction of semantically significant phrases. In: *AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, CA 1996
- Lee, K., Rho, S.: A method of generating customer's profile without history for providing recommendation to new customers in E-commerce. In: 7th FTRA International Conference on Future Information Technology, FutureTech 2012, Vancouver, BC 2012. *Lecture Notes in Electrical Engineering*, pp. 83-88. Elsevier
- Lenz, M.: Case Retrieval Nets Applied to Large Case-Bases. In: 4th German Workshop on CBR, Humboldt-Universität zu Berlin 1996, pp. 111-118. *Informatik Preprints*
- Liu, N.-H.: Comparison of content-based music recommendation using different distance estimation methods. *Appl. Intell.* doi: **10.1007/s10489-012-0363-y** (2012). doi:10.1007/s10489-012-0363-y
- Marin, L., Isern, D., Moreno, A., Valls, A.: On-line dynamic adaptation of fuzzy preferences. *Inf. Sci.* **220**, 5-21 (2013). doi:10.1016/j.ins.2011.10.008
- Marin, L., Merigó, J.M., Valls, A., Moreno, A., Isern, D.: Induced Unbalanced Linguistic Ordered Weighted Average. In: Galichet, S., Montero, J., Mauris, G. (eds.) 7th conference of the European Society for Fuzzy Logic and Technology (EUSFLAT-2011) and LFA-2011, Aix-les-Bains, France 2011a, pp. 1-8. Atlantis Press
- Marin, L., Moreno, A., Isern, D.: Automatic learning of preferences in numeric criteria. In: Fernandez, C., Geffner, H., Manya, F. (eds.) *Catalan Conference on Artificial Intelligence, CCIA 2011, Lleida 2011b*, pp. 120-129. IOS Press

- Marlin, B.M.: Modeling User Rating Profiles for Collaborative Filtering. Paper presented at the Neural Information Processing Systems (NIPS 2003), Vancouver and Whistler, British Columbia, Canada,
- Marrow, P., Hanbidge, R., Rendle, S., Wartena, C., Freudenthaler, C.: MyMedia: Producing an Extensible Framework for Recommendation. In: NEM summit 2009, Saint Malo, France 2009, pp. 28-30
- Martínez, L., Barranco, M.J., Pérez, L.G., Espinilla, M.: A Knowledge Based Recommender System with Multigranular Linguistic Information. In: Li, T., Xu, Y., Ruan, D. (eds.) International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2007, Chengdu, China, October 15-16 2007. Advances in Intelligent Systems Research. Atlantis Press
- Merigó, J.M., Casanovas, M.: Decision making with distance measures and linguistic aggregation operators. *International Journal of Fuzzy Systems* **12**(3), 190-198 (2010)
- Merigó, J.M., Casanovas, M.: Decision-making with distance measures and induced aggregation operators. *Computers and Industrial Engineering* **60**(1), 66-76 (2011a)
- Merigó, J.M., Casanovas, M.: The uncertain generalized OWA operator and its application to financial decision making. *International Journal of Information Technology & Decision Making* **10**(2), 211–230 (2011b). doi:10.1142/S0219622011004300
- Merigó, J.M., Casanovas, M.: The uncertain induced quasi-arithmetic OWA operator. *Int. J. Intell. Syst.* **26**(1), 1-24 (2011c)
- Mianowska, B., Nguyen, N.T.: Tuning user profiles based on analyzing dynamic preference in document retrieval systems. *Multimedia Tools and Applications* **65**(1), 93-118 (2013)
- Minio, M., Tasso, C.: User Modeling for Information Filtering on Internet Services: Exploiting an Extended Version of the UMT Shell. Paper presented at the UM96 Workshop on User Modeling for Information Filtering on the WWW, Kailua-Kowa, Hawaii, USA,
- Montaner, M.: Collaborative Recommender Agents Based on Case-Based Reasoning and Trust. Universitat de Girona (2003)
- Montaner, M., López, B., de La Rosa, J.L.: A taxonomy of recommender agents on the internet. *Artif. Intell. Rev.* **19**(4), 285-330 (2003)
- Morales-del-Castillo, J.M., Pedraza-Jiménez, R., Rufz, A.A., Peis, E., Herrera-Viedma, E.: A semantic model of selective dissemination of information for digital libraries. *Inf. Tech. Lib.* **28**(1), 21-30 (2009)
- Morales-del-Castillo, J.M., Peis, E., Ruiz, A.A., Herrera-Viedma, E.: Recommending biomedical resources: A fuzzy linguistic approach based on semantic web. *Int. J. Intell. Syst.* **25**(12), 1143-1157 (2010)
- Moreno, A., Valls, A., Isern, D., Marin, L., Borràs, J.: SigTur/E-Destination: Ontology-based personalized recommendation of Tourism and Leisure Activities. *Eng. Appl. Artif. Intell.* **26**(1), 633–651 (2013). doi:10.1016/j.engappai.2012.02.014
- Noppens, O., Luther, M., Liebig, T., Wagner, M., Paolucci, M.: Ontology-supported Preference Handling for Mobile Music Selection. In: Junker, U., Kießling, W. (eds.) Multidisciplinary Workshop on Advances in Preference Handling in conjunction with ECAI 2006, Riva del Garda, Italy, August 28 - 29 2006, pp. 96-101
- Öztürk, M., Tsoukias, A., Vincke, P.: Preference Modelling. In: Preferences: Specification, Inference, Applications. vol. 04271.

- Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, Dagstuhl, Germany, (2006)
- Pasi, G., Yager, R.R.: Modeling the concept of majority opinion in group decision making. *Inf. Sci.* **176**(4), 390-414 (2006). doi:DOI: 10.1016/j.ins.2005.07.006
- Passuello, A., Schuhmacher, M., Mari, M., Cadiach, O., Nadal, M.: A spatial multicriteria decision analysis to manage sewage sludge application on agricultural soils. In: Olej, V., Obršalová, I., Krupka, J. (eds.) *Environmental Modeling for Sustainable Regional Development: System Approaches and Advanced Methods*. pp. 221-241. (2011)
- Pazzani, M., Billsus, D.: Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learn.* **27**(3), 313-331 (1997). doi:10.1023/a:1007369909943
- Pham, M.Q.N., Nguyen, M.L., Ngo, B.X., Shimazu, A.: A learning-to-rank method for information updating task. *Appl. Intell.* **37**(4), 499-510 (2012). doi:10.1007/s10489-012-0343-2
- Phelan, O., McCarthy, K., Bennett, M., Smyth, B.: On using the real-time web for news recommendation & discovery. In: 20th international conference companion on World wide web, WWW 2011, Hyderabad, India 2011a. ACM New York, NY, USA
- Phelan, O., McCarthy, K., Bennett, M., Smyth, B.: Terms of a feather: content-based news recommendation and discovery using Twitter. In: Clough, P., Foley, C., Gurrin, C., Lee, H., Jones, G.J.F. (eds.) 33rd European conference on Advances in information retrieval, ECIR 2011, Dublin, Ireland 2011b. LNCS, pp. 448-459. Springer Verlag
- Phelan, O., McCarthy, K., Smyth, B.: Buzzer - online real-time topical news article and source recommender. In: Coyle, L., Freyne, J. (eds.) 20th Irish Conference on Artificial Intelligence and Cognitive Science, AICS 2009, Dublin, Ireland 2009. *Lecture Notes in Computer Science*, pp. 251-261. Springer
- Pijuan, J., Valls, A., Passuello, A., Schuhmacher, M.: Evaluating the impact of sewage sludge application on agricultural soils. Paper presented at the XV Congreso español sobre tecnologías y lógica fuzzy (ESTYLF), Huelva (Spain), February, 2010
- Porcel, C., Herrera-Viedma, E.: Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries. *Know.-Based Syst.* **23**(1), 32-39 (2010)
- Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B.: User satisfaction in long term group recommendations. In: Ram, A., Wiratunga, N. (eds.) 19th International Conference on Case-Based Reasoning, ICCBR 2011, London, UK, September 12-15 2011. *Lecture Notes in Artificial Intelligence*, pp. 211-225. Springer Verlag
- Quijano-Sánchez, L., Recio-García, J.A., Díaz-Agudo, B., Jiménez-Díaz, G.: Social Factors in Group Recommender Systems. *ACM Transactions on Intelligent Systems and Technology* **in press** (2012)
- Rastegari, H., Shamsuddin, S.M.: Web search personalization based on browsing history by artificial immune system. *International Journal of Advances in Soft Computing and its Applications* **2**(3), 282-301 (2010)
- Resnick, P., Varian, H.R.: Recommender Systems. *Commun. ACM* **40**(3), 56-58 (1997)

- Ricci, F., Rokach, L., Shapira, B.: Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook. pp. 1-35. Springer US, (2011)
- Rich, E.: User Modeling via Stereotypes. *Cognitive Science* **3**, 329–354 (1979)
- Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. SIGIR Forum (ACM Special Interest Group on Information Retrieval), 253-260 (2002)
- Serrano-Guerrero, J., Herrera-Viedma, E., Olivas, J.A., Cerezo, A., Romero, F.P.: A google wave-based fuzzy recommender system to disseminate information in University Digital Libraries 2.0. *Inf. Sci.* **181**(9), 1503-1516 (2011). doi:10.1016/j.ins.2011.01.012
- Sigurbjörnsson, B., Van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: 17th international conference on World Wide Web, WWW 2008 2008, pp. 327-336. ACM New York
- Sorensen, H., McElligot, M.: PSUN: A Profiling System for Usenet News. In: CKIM 95 Workshop on Intelligent Information Agents, Baltimore, US 1995
- Sun, M., Li, F., Lee, J., Zhou, K., Lebanon, G., Zha, H.: Learning multiple-question decision trees for cold-start recommendation. Paper presented at the Proceedings of the sixth ACM international conference on Web search and data mining, Rome, Italy,
- Symeonidis, P., Nanopoulos, A., Manolopoulos, Y.: A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *IEEE Trans. Knowl. Data Eng.* **22**(2), 179-192 (2010). doi:10.1109/TKDE.2009.85
- Torra, V., Narukawa, Y.: Modeling Decisions: Information Fusion and Aggregation Operators. Cognitive Technologies. Springer Berlin Heidelberg, (2007)
- Tso, K.H.L., Lars, S.-T.: Empirical Analysis of Attribute-Aware Recommender System Algorithms Using Synthetic Data. *Journal of Computers* **1**(4), 18-29 (2006)
- Valls, A., Batet, M., López, E.M.: Using expert's rules as background knowledge in the ClusDM methodology. *Eur. J. Oper. Res.* **195**(3), 864-875 (2009). doi:10.1016/j.ejor.2007.11.019
- Valls, A., Pijuan, J., Schuhmacher, M., Passuello, A., Nadal, M., Sierra, J.: Preference assessment for the management of sewage sludge application on agricultural soils. *International Journal of Multicriteria Decision Making* **1**, 4-24 (2010). doi:10.1504/ijmcdm.2010.033684
- Weakliam, J., Bertolotto, M., Wilson, D.: Implicit interaction profiling for recommending spatial content. In: Shahabi, C., Boucelma, O. (eds.) 13th ACM International Workshop on Geographic Information Systems, GIS 2005, Bremen, Germany 2005, pp. 285-294. ACM Press
- Wei, G., Zhao, X., Lin, R.: Some induced aggregating operators with fuzzy number intuitionistic fuzzy information and their applications to group decision making. *International Journal of Computational Intelligence Systems* **3**(1), 84-95 (2010)
- Xia, M., Xu, Z.: Entropy/cross entropy-based group decision making under intuitionistic fuzzy environment. *Inf. Fusion* **13**(1), 31-47 (2012). doi:10.1016/j.inffus.2010.12.001
- Xu, Z.: Induced uncertain linguistic OWA operators applied to group decision making. *Inf. Fusion* **7**(2), 231-238 (2006)

- Xu, Z.: Linguistic aggregation operators: An overview. *Studies in Fuzziness and Soft Computing* **220**, 163-181 (2008)
- Xu, Z.: An interactive approach to multiple attribute group decision making with multigranular uncertain linguistic information. *Group Dec. Negotiation* **18**(2), 119-145 (2009)
- Xu, Z.S., Da, Q.L.: An Overview of Operators for Aggregating Information. *Int. J. Intell. Syst.* **18**(9), 953-969 (2003)
- Yager, R.R.: On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Trans. Syst. Man Cybern.* **18**(1), 183-190 (1988). doi:10.1109/21.87068
- Yager, R.R.: Ordinal measures of specificity. *International Journal of General Systems* **17**(1), 57-72 (1990). doi:10.1080/03081079008935096
- Yager, R.R.: Including importances in OWA aggregations using fuzzy systems modeling. *IEEE Trans. Fuzzy Syst.* **6**(2), 286-294 (1998)
- Yager, R.R.: Measures of specificity over continuous spaces under similarity relations. *Fuzzy Sets Syst.* **159**(17), 2193-2210 (2008)
- Yager, R.R.: On the dispersion measure of OWA operators. *Inf. Sci.* **179**(22), 3908-3919 (2009). doi:DOI: 10.1016/j.ins.2009.07.015
- Yager, R.R., Filev, D.P.: Induced Ordered Weighted Averaging operators. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **29**(2), 141-150 (1999)
- Yong, D., Wenkang, S., Feng, D., Qi, L.: A new similarity measure of generalized fuzzy numbers and its application to pattern recognition. *Pattern Recognit. Lett.* **25**(8), 875-883 (2004)