García García, I.; Pajares Ferrando, S.; Sebastiá Tarín, L.; Onaindia De La Rivaherrera, E. (2012). Preference elicitation techniques for group recommender systems. Information Sciences. 189:155-175. doi:10.1016/j.ins.2011.11.037.

Preference elicitation techniques for group recommender systems

I. Garcia*, S. Pajares, L. Sebastia, E. Onaindia

Universitat Politècnica de València, Camino de Vera s/n, E46022-Valencia (Spain)

Abstract

A key issue in group recommendation is how to combine the individual preferences of different users that form a group and elicit a profile that accurately reflects the tastes of all members in the group. Most group recommender systems (GRSs) make use of some sort of method for aggregating the preference models of individual users to elicit a recommendation that is satisfactory for the whole group. In general, most GRSs offer good results, but each of them have only been tested in one application domain.

This paper describes a domain-independent GRS that has been used in two different application domains. In order to create the group preference model, we select two techniques that are widely used in other GRSs and we compare them with two novel techniques. Our aim is to come up with a model that weighs the preferences of all the individuals to the same extent in such a way that no member in the group is particularly satisfied or dissatisfied with the final recommendations.

Keywords: Group recommender systems, group profile, preference elicitation.

1. Introduction

A **Recommender System** (RS) [52] is a widely used mechanism for providing advice to people in order to select a set of items, activities or any other kind of product. Typically, RSs are designed to provide recommendations for a single user taking the user's interests and tastes into account. These systems are particularly well adapted to e-commerce applications where users need to be guided through complex product spaces [55] because RSs help to overcome the information overload problem. Examples include: *MyGROCER* [34] or *Buyingnet* [33]. Apart from e-commerce applications, RSs have been applied to other domains, such as the recommendation of tourist attractions (e.g. *COMPASS*

*Corresponding author

Email addresses: ingarcia@dsic.upv.es (I. Garcia), spajares@dsic.upv.es (S. Pajares), lstarin@dsic.upv.es (L. Sebastia), onaindia@dsic.upv.es (E. Onaindia)

[62], DieToRecs [17] and ITR [53]), or the recommendation of movies (MovieLens [43]).

While many RSs are focused on making recommendations to a single user, many daily activities such as watching a movie or going to a restaurant involve a group of users [11], in which case recommendations must take into account the tastes and preferences of all the users in the group [3]. This type of system is called a **Group Recommender System** (GRS). The main issue in group recommendation is to identify the items that are likely to satisfy all of the group members adequately [28, 49]. By taking into account the preferences of the group as a whole, GRSs are capable of finding a compromise that is acceptable to all the members in the group.

Over the last few years, GRSs have been an active area of research within the field of RS. As a result, some remarkable GRSs have been developed[1]. For example, *Polylens* [45] recommends movies, as an extension of the MovieLens recommender; *MusicFX* [41] selects a radio station among 91 stations of commercial-free music, each representing a different musical genre; *Intrigue* [3], *CATS* [42] and *Travel Decision Forum* [28, 29] deal with a tourist domain (the area around Torino city, skiing vacations, and the selection of a vacation destination, respectively).

The computation of accurate group models is a crucial point in group recommendation because the way in which the specification and elicitation of the users' preferences are managed in order to come up with the group model will determine the success of the system [28, 49]. GRSs usually define quite simple aggregation mechanisms to elicit the group model [39, 40], such as *Average* and *Average without Misery*. *Average* is a mechanism that aggregates the users' preferences and rates each preference as the average value of the ratings given by the users to the preference. *Average without Misery* only assigns numerical ratings to the preferences that are shared by all the members in the group but without those individual preferences that score below a certain threshold. Despite their simplicity, existing GRSs that implement these techniques have reported good results [39, 40]. Moreover, [40] details some experiments with real users in order to determine which aggregation mechanism performs best. Specifically, each user was asked to rate a recommendation for an artificial group. The result was that the preferred recommendations were those obtained by using *Average* or *Average without Misery* techniques.

For this reason, we have selected these techniques to perform some experiments to emphasize their strengths and weaknesses when applied to two different domains: a tourism domain and a movies domain. To the best of our knowledge, current state-of-the-art GRSs are not general-purpose as they are only applicable to one specific domain, and these GRSs usually contain some domain-specific aspects to improve the recommendation. Therefore, we are interested in investigating whether a general implementation of these techniques (without taking into account the specific domain where the GRS is going to be

[1]These and other systems will be introduced in section 2.

applied), would obtain as good results as expected.

As shown in section 5, from the analysis of the *Average* and the *Average without Misery* techniques, we have concluded that the *Average without Misery* technique might not be able to provide enough items that satisfy the requirements of the group; and that the *Average* technique might not be able to satisfy all the group members equally.

This has led us to develop two novel *incremental* mechanisms for preference management; *Incremental Intersection* and *Incremental Collaborative Intersection* are introduced as a way of alleviating the drawbacks of the *Average* and the *Average without Misery* strategies. All in all, the aim of this paper is to investigate which preference elicitation strategies tend to favour an equal weighting of the individual preferences when recommending an item for the group such that no member is particularly dissatisfied with the decisions. The experimental results show that these techniques behave differently according to the type of domain and the group size.

In this paper, we describe a GRS that can be used with any ontology-based application domain as well as with several group modelling strategies. The main contribution of this paper is the description of four preference elicitation mechanisms for group modelling and their performance in two application domains: a tourism domain and a movies domain.

The paper is organized as follows. Section 2 presents a taxonomy to classify GRSs and describes some of the most relevant GRSs. Section 3 provides an overview of the functioning of our system and presents the information required for the recommendation process in the domains where we have tested our GRS, namely a movies domain and a tourism domain. Section 4 explains in detail the steps of the recommendation process, which first elicits the individuals' preference model from the users' profiles and then creates a set of group preferences that is labeled with a degree of interest. Section 5 presents the group modelling strategies; first, the *Average* and the *Average without Misery* strategies are described and then we introduce the two incremental techniques. Section 6 shows the experimental results obtained when testing these techniques in the two aforementioned domains, and, finally, we conclude and present some further work in section 7.

2. Related work

This section is devoted to studying the most relevant GRSs (a description of other GRSs can be found in [11]), namely *Intrigue* [3], *Polylens* [45], *MusicFX* [41], *Let's Browse* [36], *The Collaborative Advisory Travel System, CATS* [42] and *Travel Decision Forum* [28, 29]. We classify these GRSs into categories based upon the different features used in the design of the GRSs [15, 40]. The features that characterize GRSs include the method of acquiring information about the group preferences, the process for generating recommendations, and the methods used to reach consensus among the group members [30]. Table 1 presents a classification of the aforementioned GRSs based on six independent features that heavily influence the design of GRSs:

	Intrigue	Polylens	MusicFX	Let's Browse	CATS	Travel Decision Forum	Our GRS
Information source	S CB	S CF	S CB	S CF	S CB	S CB	H
User interaction	PM	PM	PM	PM	AM	AM	PM
Domain	DS	DS	DS	DS	DS	DS	DG
Outcome	LR	LR	SR	SR	SR	P	LR
Group size	SGS	UGS	UGR	SGS	SGS	SGS	UGS
Aggregation criterion	AR	AR	AP	AP	AP	AP	AP

Table 1: GRS comparison. S-simple, H-hybrid, CB-content-based, CF-collaborative filtering, PM-passive members, AM-active members, DS-domain-specific, DG-generalist, LR-list of recommendations, SR-single recommendation, LP-list of preferences, SGS-small group size, UGS-unlimited group size, AR-aggregating recommendations, AP-aggregating profiles.

1. **Information source**. Basically, we can distinguish between **simple** approaches and **hybrid** approaches. The main simple approaches [1] are content-based (CB) [8, 10, 24, 44, 48, 61] and collaborative filtering (CF) [14, 23, 31, 35, 51, 57, 64], but there are some others like demographic [46, 67] or knowledge-based [66]. Hybrid approaches [46] result from the combination of two or more simple approaches. Most GRSs follow a CB approach, like *MusicFX* and *Travel Decision Forum*, and others, such as *Polylens*, are based on CF.
2. **User-system interaction** [15, 40]. Individuals can be dichotomized into **passive members** and **active members**. For active members, the final purpose is to reach a consensus among the group members by evaluating the system recommendations. In contrast, when members are passive, the final purpose is simply to provide a recommendation to the group without further user interaction with the system. For instance, members are active in *Travel Decision Forum* and *CATS*. Moreover, systems with passive members can also be categorized according to the amount of data that the GRS needs to make a recommendation. Thus, *Let's Browse* and *Intrigue* require little information from the user, whereas *Polylens* and *MusicFX* need the user to introduce many more details.
3. **Domain**. This feature is related to the type of domains that the GRS can work with. We distinguish between **domain-specific** GRSs and **generalist** GRSs. To the best of our knowledge, all the aforementioned GRSs are only able to work on a specific domain. *Intrigue*, *CATS* and *Travel Decision Forum* deal with a tourist domain (the area around Torino city, skiing vacations, and the selection of a vacation destination, respectively). *PolyLens* recommends movies as an extension of the *MovieLens* recommender, whereas the goal of *MusicFX* is to select a radio station and *Let's*

Browse is an agent that assists a group of people in web browsing.

4. **Outcome of the GRS** [40]. GRSs like *MusicFX*, *Let's Browse*, or *CATS* return a single recommendation; therefore, the outcome must be a successful selection for the group. Other GRSs return an ordered list of items, and the system selects among all the available items those that best match the group members, ordering the items according to a social preference function. This is the case, for instance, of *Intrigue* and *PolyLens*. As a special case, it is interesting to note that *Travel Decision Forum* does not return recommendations, but a list containing the group preferences.
5. **Group size**. *Intrigue* and *MusicFX* are capable of working with any group size[2]. *Polylens* and *Let's Browse* accept rather small groups of two or three users. The experiments in *CATS* were carried out with groups of four users.
6. **Aggregation approach to make the recommendations** [15]. GRSs are classified into (1) the ones that make the *group recommendation out of the individuals' recommendations*, and (2) those that *merge the profiles of multiple people and create a single group preference model*.
   Systems that create group recommendations by **aggregating the individual recommendations** use a two-step strategy; first, the system elicits a recommendation for each group member, and then it computes the group recommendation by adding the individual recommendations into a single list. This is done by applying a social value function that measures the appropriateness of each individual recommendation for the group as a whole. This approach is followed, for instance, by *Intrigue* and *Polylens*. *Intrigue* uses the *Weighted Average* strategy [39], splitting the group into subgroups of users with similar characteristics, such as children or disabled people. The group recommendation is based on the preferences of subgroups, the number of people in the subgroup, and the subgroup's relevance. *Polylens* uses the *Least Misery* strategy [39], i.e., the weight of a movie is the minimum of its weights in the individual recommendations. Thus, a group is as satisfied as its least satisfied member.
   In contrast, other systems build up a group preference model by **aggregating the individual preferences**. This type of group profile can be explicitly built if users use a common group account to express their preferences, or implicitly, by aggregating the different individuals' profiles or preferences. Thus, *MusicFX* and *Let's Browse* build the group profile as the result of the application of a social value function. Specifically, *MusicFX* uses the *Average Without Misery* strategy to create a group preference model (which consists of a list of music genres), thus considering only the individual ratings of each music genre that score above a certain threshold. *Let's Browse* first creates the individual profiles as a set of about 50 keyword-weighted pairs obtained by a crawler from the user's page. Then the group preference model results from a simple linear

[2] A limit in the group size has not been specified.

combination of the individuals' profiles.
GRSs like *CATS* and *Travel Decision Forum* build the group profile by means of a conversational mechanism or by critiquing the recommendations. *Travel Decision Forum* uses an interactive negotiation mechanism among agents that represent the group members and with a mediator agent that is in charge of proposing a value for each attribute in the group preference model. *Travel Decision Forum* implements multiple aggregation strategies such as Average and Median strategies. *CATS* maintains a group model by combining the individual models and associating critiques with the users who contributed them. *CATS* recommends an item to a user in the context of a group, taking into account both the individual and the group preference model to make the recommendation. If the user is satisfied with the recommendation, then it is shown to the whole group for their consideration. While users indicate which features a recommended item needs to have through critiquing, the group model contains the requirements of all users, and the item that fulfills the most requirements is recommended.

Other RSs regard user preferences as a set of constraints and attempt to find recommendations that fit the constraints of all of the users [38]. However, these systems tend to eliminate items that would cause conflict [7].

3. Our Group Recommender System

This section explains the main characteristics of our GRS and the information required in the recommendation process. The last column of Table 1 shows the classification of our GRS following the taxonomy introduced in section 2.

Unlike the GRSs described in the previous section, our proposal is a **domain-independent GRS** [21] where the set of items to recommend are determined by the particular domain ontology used by the GRS. In this paper, we will describe the GRS through the use of a tourism domain and a movies domain, but our GRS is able to work with any application domain provided that items are described by an ontology. Moreover, the system is also able to deal with any group size.

Our GRS builds a group profile by aggregating the preferences of the group members. The group recommendation is done by following a three-stage process:

1. We elicit the preference model of each group member through a hybrid approach that mixes different basic recommendation techniques (Demographic + CF + CB + General-Likes Filtering).
2. The outcome of the hybrid technique, the users' preference models, are the input data for creating the group preference model (or group profile). At this stage, the GRS uses several strategies to aggregate the individual preference models.
3. The group profile is used to select the list of items to recommend.

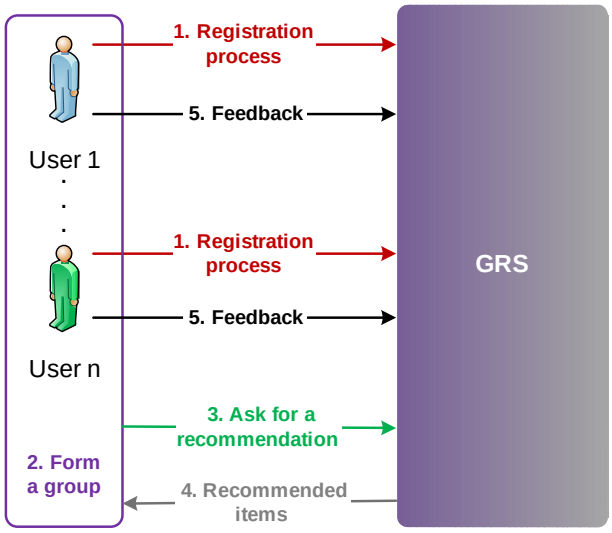

Figure 1: Steps in the recommendation process from the user point of view.

The GRS incorporates several preference aggregation strategies even though only one of them can be used at a time. Ours is a fully configurable GRS that can be set up to run any aggregation strategy or recommendation technique [21]. Therefore, including a new strategy or technique into the GRS is as easy as developing a new module.

Our GRS works with passive members because once they introduce their individual profiles, there is no further interaction with the system until the list of recommended items is returned. Specifically, **from the users' point of view**, the steps of the recommendation process are (Figure 1):

1. A person who wants to use the GRS for the first time has to **register** and (optionally) enter his/her personal details, such as name, age, gender, family, etc. The acquisition of data related to the general likes of the person in the specific application domain[3] is also necessary. For example, the user might be interested in *Science Museums* or *Parks* in a tourism domain, or in film genres like *Comedy*, *Thriller*, or *Musical* in a movies domain. Personal details and the user's general likes, which reflect the user's preferences on the domain characteristics, make up the **user profile**, which is updated accordingly as long as the user interacts with the system after the recommendation process. The components of the user profile are detailed in section 3.2. If the user is already registered, his/her profile will be available in the GRS and this step can be skipped.

[3]These data can be introduced by the user, as in the tourism domain, or automatically extracted from the available data pool, as in the movies domain (details on data acquisition are explained in section 3.2).

2. When a group of users whose members are registered in the system wants a recommendation for the group, they have to explicitly indicate that they **form a group**. Currently, our GRS only supports *ephemeral or occasional groups* [12, 45], i.e., non-permanent groups that are occasionally formed for a single recommendation.
3. The group requests a recommendation from the GRS, indicating the number of items (N) they want to obtain as a result[4]. The process to elicit the recommended items is the **GRS recommendation process**, which is explained in section 4.
4. The GRS returns a **list of N recommended items** to the group.

A crucial aspect in any RS is the **user feedback**, as a mechanism to learn about a particular user. In our GRS, we give each group member the opportunity to express his/her opinion about the recommendation. After examining the recommended items (watching a movie, visiting a place, etc.), the users will individually rate the proposed items. A given recommendation may please only some of the users while the rest of the members may have a different perception on the satisfaction reported by the recommendation. The system will use this feedback to increase the accuracy of the individuals' profiles and thus make better recommendations of the items that are likely to be of interest to the users in future interactions. The feedback provided by the users as members of the group is used to update their individual profiles rather than the group profile as we are working with ephemeral groups. By refining the profile of a group member, the system improves future recommendations for a group in which this user participates.

3.1. Domain Ontology

Our system relies on the use of a domain ontology [25] to describe the user's likes and the items to recommend in the particular domain. Recently, some researchers have been focusing on enhancing recommendations by exploiting a semantic description of the domain in which recommendations are provided [63, 65]. In general, items handled by the system are semantically described through an ontology and recommendations are based on the semantic matching between the user's profile and the item description. A limitation of this approach is that a semantic representation of the domain must be available and, until now, users' profiles and items descriptions have been manually supplied. However, the work in [54] shows some techniques for automating the process of associating features to items (as automatic content analysis [5, 47]).

The entities of the **domain ontology** in our GRS are arranged in a hierarchical structure that is connected through an *is-a* relationship in which the classification levels become more specific towards the bottom (see Figures 2 and 3). Classes in the ontology represent the **features** (F) that are commonly managed in the corresponding domain. Examples of features in the movies domain

[4]If the group does not indicate the number of recommendations required, a default value will be used.

(Figure 2) are *Comedy*, *Drama*, and *Romance*. The leaf nodes of the ontology (instances of classes) represent the **items** to recommend (*Cinema Paradiso* or *Chairman of the Board* in Figure 2).

The edges that link an item to a feature are associated to a value that indicates the **degree of interest** of the item under the corresponding feature, i.e., as a member of the category denoted by the feature. The degree of interest of the item i under the feature f (d^{if}) is the degree of suitability of the item to the feature. **Items** are described by means of a set of tuples which represent all the incoming edges of a leaf node. A tuple that describes an item i is of the form (f, d^{if}), where $f \in F$ is a feature defined in the ontology such that there is an edge connecting f and the item i, and $d^{if} \in [0, 100]$ is the degree of interest of the item i within the category represented by the feature f. For example, in Figure 2, the item *Cinema Paradiso* under the feature *Comedy* is represented by the tuple *(Comedy, 40)*, and the item *Chairman of the Board* is described by *(Comedy, 80)*, thus indicating that the latter movie is more fun than *Cinema Paradiso*.

Additionally, an item i is associated to a numeric value RC^i (**positively rated counter**). This value is increased each time a user positively rates the item i during the feedback process. RC^i represents how popular the item i is among all of the users involved in the computation of RC^i, that is, among all of the users who have been recommended and have rated the item i at any time. This value is used to distinguish the ones with the highest interest among the items classified into the same category. For example, in Figure 2, the items *Cinema Paradiso* and *Titanic* are both classified as *Romance* with the same degree of interest ($d^{CinemaParadiso,Romance}$=40, $d^{Titanic,Romance}$=40). If the item *Titanic* has a positive rating $RC^{Titanic} = 99$ and the item *Cinema Paradiso* has a positive rating $RC^{CinemaParadiso} = 60$, then the GRS will first recommend *Titanic* to users who are interested in *Romance*.

As explained above, our GRS has been designed to be able to work with any application domain provided that the domain information is specified through a domain ontology [21]. The following sections describe the ontologies of the two selected domains, movies and tourism[5].

3.1.1. Ontology of the Movies domain

MovieLens[6] is a well-known **movie database** that was created by the GroupLens research group at the University of Minnesota. Figure 2 depicts part of the movies ontology. Movies are described by a set of film genres, which are the features in our ontology. However, the data in MovieLens lack some of the information that we require for the recommendation process, namely the degree of interest of items under the domain features and the positively rated counter. For this reason, we applied some processing on the movies database in order to

[5]These sections describe the information stored in the recommender system database and how this info is obtained for the domains under consideration.

[6]http://www.grouplens.org/

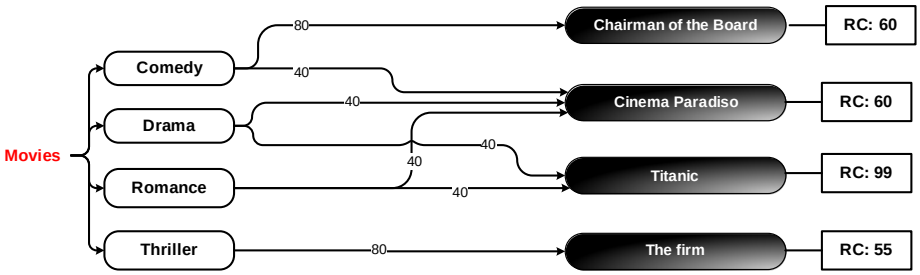

Figure 2: Part of the movies ontology.

compute these values.

Besides the classification of movies into genres, the MovieLens data set also contains the movie ratings of the users. From this information, the GRS calculates the remaining data it needs for the recommendation:

1. The **degree of interest** of a movie i under a feature f (d^{if}) is computed from the ratings of movie i and the preference of each user for the film genre f. This preference is obtained (as explained in section 3.2.1) by taking into account the ratings that u has given to movies classified under f. Depending on the number of individual ratings that a movie has, different weighting is attached to the calculated value. The weight reflects the confidence in the degree of interest of a movie; the higher the number of users, the more confidence in d^{if}.
2. The **positively rated counter** of each movie i (RC^i) is the number of users who have positively rated the movie i with a rating above a certain threshold[7].

In Figure 2, the numbers labeling the edges represent the degree of interest of the movies under the corresponding features. The number inside the box attached to the movie is the positively rated counter of the movie. For example, we have obtained that the item *Cinema Paradiso* is described by the tuples {*(Comedy, 40), (Drama, 40), (Romance, 40)*} and the $RC^{CinemaParadiso}$ is 60.

3.1.2. Ontology of the Tourism domain

The tourism data set is a domain that was specifically created to test our GRS. It contains information about leisure and tourist activities in the city of Valencia (Spain). The ontology comprises information about architectural styles, types of buildings, historic monuments, outdoor activities, open spaces, etc. Figure 3 shows a partial view of the ontology; the item *Valencia Cathedral* is described by the tuples *(Religious Building-Church, 90)* and *(Art-Gothic, 80)*, where *Religious Building*, *Church*, *Art* and *Gothic* are features of the ontology,

[7]Specifically, ratings in MovieLens are values within the range $[1, 5]$, and we only considered ratings of value 5, thus counting 20338 scores out of a total of 96103 ratings.

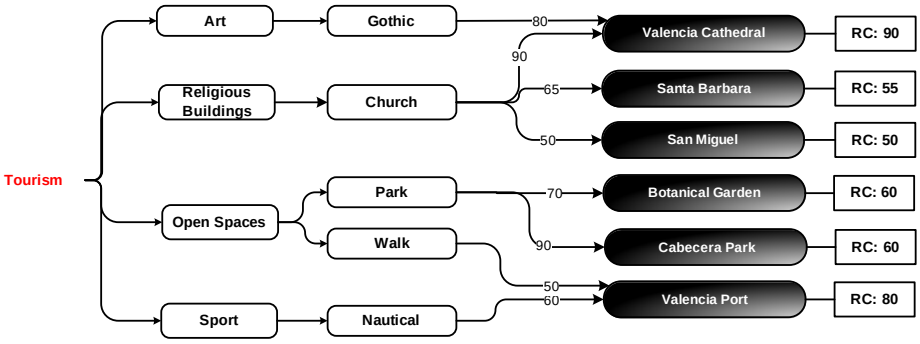

Figure 3: Part of the tourism ontology.

with *Religious Building-Church* and *Art-Gothic* being specializations of the upper features. The numeric values in the tuples represent the degree of interest of the items under the corresponding features, thus indicating that this visit is very appealing for people who are interested in religious art, particularly in gothic architecture.

The tourist data set has been extracted from the *Conselleria de Turismo de Valencia*[8]. Specifically:

1. The **classification of items under the domain features** and consequently, the **degree of interest** of an item i under a feature f (d^{if}), is not available in the tourism data set. We have obtained them through the application of an automatic web mining process.
2. The **positively rated counter** is calculated during the interaction of the users with the GRS by using the feedback generated in the recommendation process. That is, when a user rates an item with a value above a given threshold[9], the RC^i is increased.

3.2. User profile

For each user, the GRS records a **profile** [48] that contains the personal details and general likes besides the feedback acquired through the historical interaction of the user with the system. All this information can be modified later upon the request of the user.

The **profile** of a given user u stores the following information:

1. **Personal and demographic details** like age, gender or country. The user information required during the registration process depends on the domain that the GRS is working with.

[8] http://www.turisvalencia.es/
[9] This threshold is set to 80.

2. The **general-likes model** of the user (GL^u) represents the user's preferences with regard to the domain of application. It is a list of the features in the ontology which the user u is interested in, together with the numerical ratings that the user has given to those features[10]. More formally, GL^u is represented by a list of tuples of the form $GL^u = \{(f, r^{uf})\}$, where $f \in F$ is the domain feature, and $r^{uf} \in [0, 100]$ is the rating that is computed or given by the user u to the feature f[11]. The number of features in GL^u depends on the domain design. In the case of simple user-introduced ontologies, the user may be requested to rate all of the features. In more sophisticated ontologies, it suffices for the user to rate a subset of the domain features. The more features in GL^u, the more information about the user and, therefore, the more accurate the recommendation. In short, the set of preferences in the general-likes model of a user is a subset of the ontology features. As an example, we can have a user with $GL^{u_1} = \{(Thriller, 95), (Comedy, 40)\}$ in the movies domain, and $GL^{u_2} = \{(OpenSpaces, 25), (Art, 87)\}$ in the tourism domain.
3. Information feedback acquired through the **historical interaction of the user with the GRS**, namely the degree of satisfaction of the user with the provided recommendations. From the users' point of view, the feedback process is the last stage in the recommendation process. The information acquired at this stage will be further used to better capture the users' likes and update their individual profiles. As explained above, the reason why the feedback process is individually done for each user is because our GRS only supports ephemeral groups. The purpose of this process is to capture the particular opinion of the user on the recommended items and use this feedback to refine the individual profiles.

It is important to remark that the user profile is not usually a complete and exhaustive source of information. Even if the user only introduces a few demographic details and rates a subset of the domain features, the system is still able to provide a recommendation. This is because we use a hybrid recommendation technique, meaning that we apply a combination of techniques so as to exploit the information in the user profile as much as possible. In this sense, if the user does not enter preferences in the general-likes model but introduces some demographic details, the application of a demographic technique could generate a recommendation. Therefore, the more information in the user profile, the more basic recommendation techniques are applicable and, consequently, the more accurate the recommendation is.

[10] These preferences together with their numerical rating are computed in the movies domain and introduced by the user in the tourism domain as explained below.

[11] If a feature rating in GL^u is 0, it means the user has not rated it or there is not enough information to infer f as a feature of interest for the user. A low rating implies that the user is not very interested in the feature.

3.2.1. User profile in the Movies domain

Since the MovieLens data set does not contain a full description of the users' details, the user profile extracted by our GRS will only hold some of the data that would be desirable to obtain a recommendation using our GRS.

Specifically, the **personal and demographic details** of the user in the MovieLens data set are age, gender, occupation and zip code. The **general-likes model** of a user is conceived with all the features of the simple movies ontology. Since the movies database does not contain ratings of the domain features, we designed a reverse engineering process to infer these values by using the genre of the movies rated by the user in the data set. The **historical interaction of the user with the GRS** is the set of movies rated by the user[12].

3.2.2. User profile in the Tourism domain

This domain has been explicitly created to test our GRS, so the information contained in the user profile is the information that we have considered to be more suitable for the recommendation process. The users in the tourism domain are real users, and the information in their profiles has been directly filled in by them through a web service that is used as an interface with the GRS. The **personal and demographic details** include, among others, age, gender, country and whether or not it is a family trip. The tourism ontology is organized into two levels. The features in the **general-likes model** of the user (GL^u) are the ones at the first level of the ontology (i.e., *Religious Buildings*, *Art*, *Open Spaces*, etc. in Figure 3). At the registration process, users utilize the web service to rate the tourism features that they are interested in. The **historical interaction of the user with the GRS** includes all of the items rated by the user during the feedback process.

4. GRS Recommendation process

The starting point of the **GRS recommendation process** is a group of users who wants a recommendation of N items, where N is a parameter defined by the group[13]. The aim of our GRS is to return N items so that all the group members are as equally satisfied as possible.

First, the GRS builds up the individual preference models from the users' profiles and then it elicits the group preference model through aggregation of the users' preference models. Once we have the group's preferences, the GRS uses them to elicit the recommended items. Specifically, the three steps of the GRS recommendation process are (Figure 4):

- Step 1: the *User Manager* analyzes the users' profiles and elicits the preferences for each individual. The **preference model** of an individual u

[12]Movie ratings in the MovieLens data set have been previously normalized.
[13]If the group does not indicate the number of recommendations, a default value is used.

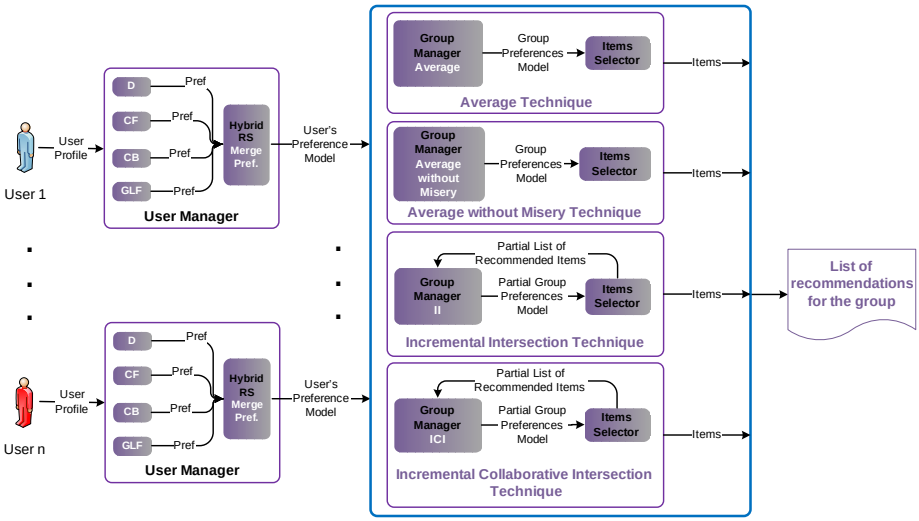

Figure 4: Steps in the recommendation process.

(P^u) is the set of preferences that characterize the person within the application domain. The output of this step is a preference model for each user in the group.

- Step 2: the *Group Manager* gathers the preference models of all the users in the group (P^u) and elicits the group preference model (P^G).

- Step 3: the *Items Selector* uses the group preference model to obtain the list of recommended items (RI^G). The items in this list are ranked according to a degree of interest of the group in each item, and the GRS returns the N highest-ranked items.

4.1. Step 1: Individual preference model

The **User Manager** analyzes the users' profiles in order to elicit the list of preferences for each user, that is, the preference model for each individual u, which we will refer to as P^u. A **preference** in this model is a tuple of the form (f, d^{uf}), where f is a feature in the ontology and $d^{uf} \in [0, 100]$ is the estimated degree of interest of the user u in the feature f [19].

P^u is computed by using a **hybrid RS** [46]. Given that Basic Recommendation Techniques (BRTs), such as CF or CB, exhibit advantages and disadvantages [2], a common solution is to combine these techniques into a hybrid RS, thus alleviating the limitations of one technique with the advantages of others and improving the recommendations [1, 6, 22, 37, 50]. Specifically, we apply a **Mixed Hybrid Technique** [14], which mixes the preferences in the user's profile through the following BRTs [19, 20]:

- A **demographic** method [46, 67], which builds a list of preferences by taking into account the user's personal and demographic details.

- A **content-based** method [8, 10, 24, 44, 48, 61], which uses the information feedback provided by the user to create new preferences from the features of the items that have been positively rated by the user.
- A **collaborative** method [14, 23, 31, 35, 51, 57, 64], which calculates the preferences from the preferences of other users who are somewhat similar to the current user.
- A **general-likes** method [26, 50], which builds an initial list of preferences with the features in GL^u.

The application of a BRT obtains a list of preferences denoted as P^u_{brt}. We refer the reader to the work in [19] for more detailed information about the process followed by each technique and how the degree of interest (d^{uf}) is computed. The preferences in each list P^u_{brt} are then merged to come up with P^u as follows:

$$P^u = \{(f, max(d^{uf})) : (f, d^{uf}) \in \bigcup_{\forall brt} P^u_{brt}\} \quad (1)$$

meaning that a feature f is included in P^u with a degree of interest which is the maximum value of the degree of interest (d^{uf}) of the feature f in all preference lists P^u_{brt}. This way, the preference model P^u keeps the set of features that are preferred by the user u. The process to elicit P^u is executed as many times as users in the group.

4.2. Step 2: Group profile

The **Group Manager** takes the preference models P^u of all group members, and elicits the **group preference model**, P^G. P^G is a set of preferences of the form (f, d^{Gf}), where f is a feature in the ontology and $d^{Gf} \in [0, 100]$ is the estimated degree of interest of the group G in the feature f. The generation of the group preference model is done with one of the following preference elicitation techniques: *Average*, *Average without Misery*, *Incremental Intersection* (*II*), or *Incremental Collaborative Intersection* (*ICI*). The incremental mechanisms (*II* and *ICI*) create the list of recommendations by successively refining the group preference model. These two algorithms are an attempt to alleviate the drawbacks found in our experiments with the *Average without Misery* and the *Average* techniques. All these operations are detailed in section 5.

4.3. Step 3: Recommendation of items

The **Items Selector**, the module in charge of selecting the items that satisfy the group's preferences (see Figure 4), uses the group preference model P^G to obtain the set of recommended items.

The **list of recommended items**, which we will call RI^G, is a set of tuples of the form $RI^G = \{(i, d^{Gi})\}$, where i is the recommended item, and d^{Gi} is the estimated degree of interest of the group G in the item i. An item described under a feature f is selected to be recommended if there is a tuple (f, d^{Gf}) in

the group preference model P^G. The degree of interest of the group G in an item i, d^{Gi}, is calculated as follows[14]:

$$d^{Gi} = \big(percentile(RC^i) + \frac{\sum_{\forall(f,d^{Gf})\in P^G}(d^{if} + d^{Gf})}{|P^G|}\big)/3 \quad (2)$$

$$\text{where,} \qquad percentile(RC^i) = \frac{RC^i}{\max(RC^i)} \times 100 \quad (3)$$

The first part of equation 2 reflects the popularity[15] of an item i among the users in the group G so that the more popular the item, the higher the value d^{Gi}. The second part of equation 2 balances the estimated degree of interest of item i according to the interest of the group in the features in P^G (d^{Gf}) and the degree of interest of item i under such features (d^{if}). Only those features included in P^G are considered in the computation of d^{Gi}.

Once the value d^{Gi} is computed for every item, the list of items are ranked according to their degree of interest. The Items Selector chooses the N best-ranked items that will form the list RI^G. It is important to note that an item that is included in the historical interaction (user profile) of at least one group member cannot be recommended for the whole group. For example, if one of the users has watched *Cinema Paradiso*, this movie cannot be recommended to the whole group.

Here is an example on how the Items Selector creates the recommendation list. Figure 3 shows some items in the tourism domain along with their features, the degree of interest of the items with regard to these features, and the $percentile(RC)$. Let's suppose that the Group Manager has elicited the following P^G for a given group:

$$P^G_{Case1} = \{(ReligiousBuildings - Church, 80), (OpenSpaces - Park, 79)\}$$

Then, the recommendations will be items described by the features *Religious Buildings-Church* and *Open Spaces-Park*. Assuming the number of requested recommendations is set to $N = 3$, the three highest-ranked items will be selected for the list RI^G_{Case1}. The estimated degree of interest of items *Valencia Cathedral* and *Cabecera Park* is calculated as follows:

- $d^{GValenciaCathedral} = (90 + (90 + 80))/3 = 87$
- $d^{GCabeceraPark} = (60 + (90 + 79))/3 = 76$

The left column in Table 2 shows the items that are finally selected as recommendations. Given that the information of P^G_{Case1} shows a similar degree of interest in *churches* and *parks*, the recommendation list comprises a visit to a

[14] This d^{Gi} is divided by 3 to obtain a value in the interval $[0, 100]$.

[15] $percentile(RC^i)$ refers to the percentile rank of the positively rated counter of i (RC^i) with respect to the whole set of items positively rated by all users who were recommended item i at any time.

Recommendation (Case 1)	Recommendation (Case 2)
(Valencia Cathedral, 87)	(Valencia Cathedral, 87)
(Cabecera Park, 76)	(Santa Barbara, 67)
(Botanical Garden, 70)	(San Miguel, 60)

Table 2: Example: items recommended for different groups.

church and to two parks. Although there are other churches, in this example, it is the factor RC^i which makes the difference in the final d^{Gi} of these items, as Figure 3 shows.

As another example, let's assume there is another group whose P^G is as follows:

$$P^G_{Case2} = \{(ReligiousBuildings - Church, 80), (OpenSpaces - Park, 10)\}$$

In this case, the degree of interest associated to the feature *Open Spaces-Park* is much lower than in Case 1. Obviously, this affects all items classified as *parks*. For example, the estimated degree of interest of the item *Cabecera Park* is $d^{GCabeceraPark} = (60 + (90 + 10))/3 = 53$ is not high enough to be selected for the final list of recommended items, as Table 2 shows.

Let us explain further the meaning of the group preference model. The group preference model P^G can be viewed in two different ways:

1. As a ranked list of preferences where the degree of interest, which is calculated using a social value function, only determines an ordering of the group's preferences regardless of the degree value [7, 32].
2. As a ranked list of preferences where the value of degree of interest also indicates the degree of preference for a particular feature; i.e., the higher the value, the more preferable the feature (see section 3).

In the first case, the social value function that is used to calculate the degree of interest and therefore to select an item aggregates scores by using users' relative preferences to search for an optimal ordering of items. The aim of this technique, which is appropriate for group decision-making, is to elicit an ordering of the items. However, this is not applicable in other GRS like ours because the value of the degree of interest of the group's preferences does not necessarily entail a strict preference ordering. For this reason, equation 2 takes into account the values assigned to each feature in P^G.

For our example, both P^G_{Case1} and P^G_{Case2} establish the same ordering between the features *Religious Buildings-Church* and *Open Spaces-Park*, but in P^G_{Case1}, the group has very similar tastes in *churches* and *parks*, whereas the second group clearly prefers *churches* over *parks*. This is reflected in the recommendations shown in Table 2. The recommendation for the first group (Case 1) contains items classified as *Religious Buildings-Church* and others classified as *Open Spaces-Park* because the degree of interest for both features in P^G_{Case1} is quite similar. However, in the second case, all the recommended items belong to the category *Religious Buildings-Church*. If we had simply considered the

User	List of Preferences P^u
u^1	(Sport-Nautical, 80) (Open Spaces-Walk, 50) (Open Spaces-Park, 30)
u^2	(Sport-Nautical, 50) (Open Spaces-Walk, 70)
u^3	(Sport-Nautical, 40) (Religious Buildings-Church, 90) (Open Spaces-Park, 70)

Table 3: Example: lists of preferences for each user in the group.

preferences as a ranking, the GRS would have only selected *churches* for the first group as it has done for the second group.

5. Mechanisms for group recommendation

The key task in our GRS is to elicit the group preference model P^G (that contains the preferences that the group is interested in) by aggregating the preference models of the group members (P^u). P^G is then used by the Items Selector to return the list of recommended items. In this section, we analyze several strategies to create P^G from the $P^u s$, namely the *Average* and the *Average without Misery* strategies, and two novel mechanisms, the *Incremental Intersection* and the *Incremental Collaborative Intersection*.

Throughout this section, we will use an example that is based on the tourism domain to illustrate the behaviour of the proposed strategies. Let's suppose that there is a group formed by three users, whose individual preferences are shown in Table 3. The number of requested recommendations for the group is $N = 3$.

5.1. Average and Average without Misery strategies

GRSs tipically use aggregation techniques to obtain a common profile for a given group. In [39], the author discusses ten different aggregation strategies to combine individuals' ratings into a single group rating. Some of these methods are: the *Average* strategy, which is used in *Travel Decision Forum*; the *Least Misery* strategy, which is used in *Polylens*; the *Average Without Misery* strategy, which is used in *MusicFX*; or the *Utilitarian* strategy, which is used in *Intrigue*. A major difference among these strategies is the emphasis that is placed on individual satisfaction compared to the satisfaction of the majority of the group, particularly on the avoidance of misery [39]; that is, a solution that leaves one or more members very dissatisfied. The work in [40] describes a series of experiments that were conducted with real users in order to determine which strategy performs best. These experiments show that the *Average* and the *Average without Misery* strategies perform best from the users' point of view because they seem to obtain similar recommendations to those that emerge from an actual discussion in a group of "humans". For this reason, we have included them in our analysis.

In this section, we briefly recall these two strategies and, in section 6, we use the results obtained with them as a basis for comparison with other more elaborated algorithms.

Mechanism	Preferences P^G	Items (degree of interest/users satisfied)
Average	(Sport-Nautical, 57) (Open Spaces-Walk,40) (Open Spaces-Park, 33) (Religious Buildings-Church, 30)	Valencia Cathedral (70/1) Cabecera Park (61/2) Valencia Port (54/3)
Average without Misery	(Sport-Nautical, 57)	Valencia Port (66/3)

Table 4: Example: Average and Average without Misery mechanisms.

5.1.1. The Average strategy

Average is one of the simplest methods for aggregating users' preferences. It simply consists in computing an average rating for each element from the individual ratings that the element receives [39].

When applying this operation in our GRS, the group preferences model, P^G, is the result of aggregating the preferences that are present in the preference model P^u of at least one member of the group. The interest of the group in a feature f, d^{Gf}, is calculated as the ***average value*** of the degree of interest of the users of G in f. Obviously, as all the features in the ontology may not be present in the users' preference models, only those features that appear in at least one P^u are considered for P^G. In addition, if a feature f is present in P^G but not in the preference model of a user u, the degree of interest of the user in the feature is set to 0 ($d^{uf} = 0$). With these ingredients in mind, the list P^G is computed as follows:

$$P^G = \{(f, d^{Gf}) : \exists (f, d^{uf}) \in \bigcup_{\forall u \in G} P^u\}, \text{where } d^{Gf} = avg(d^{uf}) \tag{4}$$

An example of the application of this strategy is shown in Table 4. The d^{Gf} value for the feature *Sport-Nautical* is calculated as the average value of d^{uf} across all group members: $d^{GSport-Nautical} = (80 + (50 + 40))/3 = 57$. Similarly, $d^{GOpenSpaces-Walk} = (50 + (70 + 0))/3 = 40$.

The final P^G is shown in the second column of Table 4. It only contains those features that are present in at least one P^u (the values d^{Gf} of the other features are considered to have a value of 0). From this P^G, the list of recommended items is shown in the third column of Table 4. In this case, although the feature *Religious Buildings-Church* has the lowest d^{Gf}, the *Valencia Cathedral* is the item that obtains the highest d^{Gi} because it is a high-valued, top-visited place in Valencia as reflected by its RC^i with a value of 90 (see Figure 3). The reason behind this is that, in our GRS, the feedback of users who have already visited a place (or watched a movie, in the movies domain), that is, the RC^i, has a large impact on subsequent recommendations, as equation 2 indicates. The second recommendation is an item that is classified as an *Open Spaces-Park*, which satisfies two of three users, whereas the last recommended item satisfies

all group members.

5.1.2. The Average without Misery strategy

This strategy makes a new list of ratings with the average rating of individual scores, but without items that score below a certain threshold for individuals [39]. In our GRS, the *Average without Misery* operation computes P^G as the set of features that are shared by all group members with a d^{uf} value above a threshold α. The group preference model is calculated as shown in equation 5:

$$P^G = \{(f, d^{Gf}) : \exists (f, d^{uf}) \in \bigcap_{\forall u \in G} P^u \wedge d^{uf} > \alpha\}, \text{where } d^{Gf} = avg(d^{uf}) \quad (5)$$

Unlike *Average*, a feature elicited with the *Average without Misery* method is only included in P^G if it appears in the list P^u of every user u in the group with a d^{uf} value greater than α. This way, only features derived by any of the BRTs (Demographic, CB, CF and General-Likes Filtering) with a d^{uf} value greater than α are considered to be of interest for the user and, therefore, recommendations with a low interest (misery) for a group member are avoided.

The last row of Table 4 shows the list of preferences computed for the group and the recommended items when applying this strategy. Assuming $\alpha = 40$, the only feature shared by all the group members is *Sport-Nautical* (see Table 3), resulting in $d^{Gf} = 57$. With this P^G, only one item can be recommended: *Valencia Port*.

5.2. Incremental algorithms for group recommendation

The examples in the previous sections demonstrate two limitations of the *Average* and *Average without Misery* strategies:

- The *Average without Misery* technique might not be able to provide enough items that satisfy the N items requested by the group. In the above example, only one recommendation is obtained with this technique.
- The *Average* technique in this case obtains N items (three items), but not all items satisfy the majority of the users.

Due to the difficulty of developing methods that accurately reflect and balance the opinions of all the members in a group [4], we propose two incremental algorithms, the **Incremental Intersection** (*II*) and the **Incremental Collaborative Intersection** (*ICI*). These two algorithms build a group preference model which is successively refined, thus allowing the list of items to recommend to be constructed incrementally. Our aim is to consider first those features which are the most satisfying for the group as a whole, and then, incrementally, take into account other less satisfying features.

Both *II* and *ICI* elicit a partial group preference model which is then used by the Items Selector to obtain the list of recommendations (Figure 4); if this list contains at least N items (the number of items requested by the group),

Algorithm 1 The Incremental Intersection algorithm.

Require: $P^{u_1}..P^{u_{|G|}}$
  Step 1. Voting Process
  Step 2. $N_{votes} = |G|$
  repeat
    Step 3. Select the features with $votes(f) == N_{votes}$
    Step 4. Select the items that match the features
    Step 5
    if Not enough items **then**
      Step 5. $N_{votes} = N_{votes} - 1$
    end if
  until Enough items

Iteration	Preferences (votes) P^G	Items (degree of interest/users satisfied)
1	(Sport-Nautical, 57) (3)	Valencia Port (66/3)
2	(Open Spaces-Walk, 40) (2) (Open Spaces-Park, 33) (2)	Cabecera Park (61/2) Botanical Garden (54/2)
	(Religious Buildings-Church, 30) (1)	

Table 5: Example: the Incremental Intersection method.

the process ends; otherwise, new preferences are added to the partial group preference model, and the Items Selector calculates a new list of recommended items. In the *II* algorithm, this extension of the P^G is done through a voting procedure whereas the *ICI* uses a collaborative recommender system to select which features should be considered. The process continues until the Items Selector finds N items for the group.

5.2.1. The Incremental Intersection Method

The aim of the *Average without Misery* method is to recommend items that satisfy all members in the group. However, this is not possible when, for example, the preference model of one member is very restrictive or far different from the others' preference models, or when dealing with big groups, where it is difficult to find a large number of common preferences. In these cases, one choice is to satisfy only some of the members of the group; however, deciding which members must be ruled out is also a complicated task. Rather than discarding members of the group, we opt for discarding preferences and not including them in the group preference model.

In order to achieve this, we first retrieve the interests that are shared by all members in the group. If this is not enough to get the N requested items, we then consider the preferences shared by the majority of group members. The aim of this process is to give more priority to the most alike members, i.e., to those who have more interests in common.

The *II* method is actually a weighted average of the most *voted* preferences among the users in the group, i.e., the preferences shared by the largest number of people in the group. The input of this algorithm are the individual preference models of the group members. This algorithm (see algorithm 1) can be decomposed in the following steps:

- **Step 1**: The algorithm starts a *voting process* [9, 13] where a feature f is *voted* by a user u if a preference over the feature f is present in the corresponding list P^u. More precisely, we define $votes(f)$ as the number of users whose P^u contains a preference over the feature f with a d^{uf} value greater than a threshold δ (to avoid misery). This simple voting process could be replaced or complemented with other more sophisticated methods such as a Borda protocol [27] or hybrid protocols [16].

- **Step 2**: We initialize the number of required votes to select a feature (N_{votes}) as the group size $N_{votes} = |G|$ in order to select the features with the highest number of votes.

- **Step 3**: We select the features with $votes(f) == N_{votes}$, where the function $votes(f)$ obtains the number of users in the group that have voted for the feature f. At the first iteration, we select the features with the highest number of votes ($N_{votes} = |G|$), so this is equivalent to the P^G obtained by the *Average without Misery* strategy. The d^{Gf} value that is associated to each feature is computed as the average d^{uf} value across all users:

$$P^G = \{(f, avg_{\forall u \in G}(d^{uf})) : votes(f) \geq N_{votes}\} \tag{6}$$

- **Step 4**: The Items Selector elicits the items described by the features contained in P^G, calculating the d^{Gi} value of the selected items according to equation 2.

- **Step 5**: If there are not sufficient items to cover the requested number of recommendations (N), we retrieve the features that have at least $N_{votes} = N_{votes} - 1$ votes at the next iteration, and so on. This way, we incrementally consider the features shared by the largest number of people in the group.

Table 5 shows an example of the recommendation process when using the *II* algorithm. At the first iteration, only one item associated to the most-voted feature, *(Sport-Nautical,57)*, is recommended, namely *Valencia Port*. This would be the only item recommended by the *Average without Misery* strategy (see Table 4) because it is the only one that satisfies all group members. As the group has requested three recommendations ($N = 3$), a second iteration includes the features with at least two votes: *(Open Spaces-Park, 33)* and *(Open Spaces-Walk, 40)*. In this case, two items are recommended, The *Cabecera Park* and The *Botanical Garden*. At this point, we already have three recommended

Algorithm 2 The Incremental Collaborative Intersection algorithm.

Require: $P^{u_1}..P^{u_{|G|}}$
  {*Step 1*}
  Step 1.a. $P^{AWM} \leftarrow$ Average Without Misery method
  Step 1.b. $P^G \leftarrow$ Select the features in P^{AWM} above a threshold δ
  repeat
    Step 1.c. Select the items that match P^G
    if Not enough items with high ratio **then**
      {*Step 2*}
      Step 2.a. Create a meta-user: $GL^{meta} \leftarrow P^G$
      Step 2.b. Collaborative RS: obtain P^{col} using the GL^{meta}
      Step 2.c. Add the highest rated preferences of P^{col} in P^G
    end if
  until Enough items

items so the process stops here; but, if another iteration were necessary, the preferences voted only by one user would be taken into account.

This example demonstrates that the incremental inclusion of a larger number of preferences shared by the majority of group members allows the *II* algorithm to overcome the difficulty of the *Average without Misery* strategy of finding the number of requested items. Moreover, *II* satisfies a larger number of users than the *Average* strategy because it takes into account not only the d^{Gf} value associated to the features, but also the number of users who share such feature. For example, the item *Valencia Cathedral* is not recommended when using *II*.

5.2.2. The Incremental Collaborative Intersection Method

As stated in previous sections, one obstacle in group recommendation is when the group members share very few preferences as this situation prevents the system from finding the number of requested recommendations. While the *II* algorithm resolves this problem by means of a voting strategy which gradually selects new features to be included in the list P^G, the *ICI* algorithm uses a collaborative technique to infer new features from the available knowledge about the group.

On the other hand, in situations where the group preference model contains too many preferences, the application of the *Average* and the *Average without Misery* strategies may not render good results because having a very large list of recommendations with similar estimated degrees of interest makes the selection of the items to recommend more difficult. The application of the *II* algorithm in these situations does not alleviate the problem either. However, the *ICI* algorithm overcomes this drawback through a collaborative technique that deduces the preferences that best represent the group. Specifically, *ICI* builds an initial group preference model P^G (step 1 in algorithm 2), which is successively extended with features that are preferred (step 2 in algorithm 2) until the requested number of recommendations is satisfied.

Iteration	Preferences P^G	Items (degree of interest/users satisfied)
1	(Sport-Nautical, 57) (Int.)	Valencia Port (66/3)
2	(Open Spaces-Park, 80) (CF)	Botanical Garden (70/2) Cabecera Park (69/2)

Table 6: Example: the Incremental Collaborative Intersection method.

The input of the *ICI* algorithm (see algorithm 2) are the users' preference models. The algorithm works in two steps:

- **Step 1: Build the initial group preferences model.**
  - **Step 1.a**: Through the application of the *Average without Misery* strategy, we compute the set of preferences shared by all group members and whose d^{Gf} value scores are above a given threshold. This set is called P^{AWM}.
  - **Step 1.b**: From the preferences in P^{AWM}, we only select those that are above a threshold δ to build the initial group preference model P^G. δ should be as high as possible in order to distinguish the best-rated preferences from the less interesting ones. The remaining preferences are also discarded because they usually represent the average value of a set of preferences that have been well-rated by some users, badly-rated by others, and fairly acceptable by most of the users.
  - **Step 1.c**: Once the list P^G is built, the Items Selector elicits the set of items to recommend along with their d^{Gi} value according to the preferences in P^G. If the number of items returned by the Items Selector is not enough to satisfy the group requirements (N), P^G is extended in the second step until the number of required items is reached.

  Let's consider the application of the *Average without Misery* strategy on the users' preference models of Table 3. The resulting group preference model is $P^G = \{(Sport - Nautical, 57)\}$. The Items Selector returns the item *Valencia Port* with a $d^{GValenciaPort} = (80 + (60 + 57))/3 = 66$, as Table 6 shows (iteration 1). As only one item is returned, we proceed with the second step to get the three requested recommendations.

- **Step 2: Extend the group preferences model if necessary.**

  The goal of the second step is to extend P^G with new preferences by using a collaborative RS.

  - **Step 2.a**: We define a **meta-user**, a user who is interpreted as an abstraction of the members in the group. Initially, the general likes of the meta-user are set to the preferences computed for the group: $GL^{meta} = P^G$.

- **Step 2.b**: A collaborative RS is applied to elicit the preferences that match the tastes of people with similar likes to the meta-user, which we refer to as P^{Col}. Preferences in this list have the form (f, d^{metaf}), where f is the feature, and d^{metaf} is the degree of interest of the meta-user in the feature f. This value is calculated by using the collaborative technique applied over the users who are similar to the members in the group.
- **Step 2.c**: We select from P^{col} the preferences whose d^{metaf} is greater than a threshold γ in order to complete the group preferences P^G. These preferences are inserted in P^G as tuples of the form (f, d^{Gf}) with $d^{Gf} = d^{metaf}$. Each time the list P^G is extended with new preferences, the group preference model comprises more information about the meta-user who is representing the group. Hence, the more information in P^G, the more accurate the result of the collaborative technique.

If the number of recommendations does not cover the group's request, the second step is performed again by applying the collaborative technique on the new meta-user preference model; and so on, until the number of requested recommendations is reached or no more new preferences are obtained from the application of the collaborative RS. This incremental procedure brings more and more preferences to the group preference model and, consequently, more recommendations can be made.

Following with the example, the initial preference model of the meta-user is $GL^{meta} = \{(Sport - Nautical, 57)\}$. Let's suppose that the preferences obtained with the collaborative RS are: $P^{col} = \{$*(Open Spaces-Park, 80), (Open Spaces-Walk, 45)*$\}$. In this case, assuming $\gamma = 70$, the only preference with $d^{Gf} \geq \gamma$ is *Open Spaces-Park*. Then, the new model is $P^G = \{$*(Sport-Nautical, 57), (Open Spaces-Park, 80)*$\}$. With this new P^G, the Items Selector is able to elicit two new items: *(Botanical Garden, 70)* and *(Cabecera Park, 69)*. The last column of Table 6 shows the final list of recommended items. Since the number of the requested recommendations has been obtained, the process ends.

6. Experimental results

This section discusses the experiments conducted to evaluate the behavior of our GRS on two different domains, a tourist domain and a movies domain. Section 6.1 explains the measures used to evaluate the GRS. Sections 6.2 and 6.3 delve into each domain description and present a comparative analysis among the techniques discussed throughout the paper. Section 6.4 analyzes the differences between the domains, and we also discuss the strengths and weaknesses of each preference elicitation technique.

6.1. Description of the measures for evaluating our GRS

Recommender systems research has used several types of measures for evaluating the quality of the recommendations offered to individuals, such as precision, recall, or mean absolute error (MAE). However, to the best of our knowledge, there is not a widely accepted measure for evaluating GRSs. For this reason, we have adapted MAE to the group recommendation context because it is the most commonly used and is the easiest to interpret directly [56] when used for evaluating RSs.

First, we define MAE^u, which gives a measure of the deviation of the recommendation for the group with respect to the estimated values for a group member on his own. Given a recommendation RI^G of N items for a group G such that $u \in G$, the mean absolute error for the user u is defined as follows:

$$MAE^u = \frac{\sum_{i=1}^{N} |d^{ui} - d^{Gi}|}{N} \tag{7}$$

where d^{ui} is the estimated degree of interest of the user u in the item i. This value is obtained by a single-user RS ([21, 60]). Therefore, MAE^u indicates how adequate the group recommendation is for user u. The lower the MAE^u is, the more accurate the group recommendation is for this user.

Unlike individual recommendations, when dealing with groups, a very important issue is to obtain recommendations that are as satisfactory as possible for all the group members, that is, the avoidance of misery. Therefore, our interest is to measure two aspects:

1. The satisfaction of the group as a whole, that is, the accuracy of the group recommendation for all the group members. This is achieved by unifying the MAE^u for each group member into a single measure; specifically, we define MAE^G as the average of MAE^u for all the members of the group. Therefore, a low MAE^G indicates that the group as a whole is highly satisfied with the recommendation.
2. The degree to which the group members are equally satisfied with the recommendation. This is achieved by calculating the standard deviation (distance) on MAE^u over all the group members, which we denote as D^G. A low distance represents that all the group members are equally satisfied. That is, this measure could be interpreted as the difference between the satisfaction of each group member with respect to the satisfaction of the other group members.

6.2. Movies domain

The experiments shown in this section were performed with data taken from the MovieLens web-based movie recommender. The data set[16] contains 900 user profiles with their respective histories of interaction with the system and a set

[16]Freely available from www.grouplens.org.

of 1682 films. The ontology-based catalogue comprises 20 features (see Figure 2).

Section 3.2.1 describes how the information contained in this data set has been adapted to form the *user profile* in our GRS. In our experiments, the average of the features in GL^u is 15 features, and the average of the rated movies included in the user historical interaction is 45.

With regard to the *items*, in the MovieLens data set, each movie has the following characteristics: the movie title, the release date, the video release data, the IMDb URL, the film genres that describe it (2 on average), and the list of ratings given by the users (57 on average). As explained in section 3.1.1, the available information in the database has been processed to obtain the missing information in the ontology: the degree of interest of each movie in each feature and the positively rated counter.

We performed some experiments with the *Average*, the *Average without Misery*, the *Incremental Intersection*, and the *Incremental Collaborative Intersection*[17] strategies by using groups of different size ranging from 2 to 9 randomly selected users. Specifically, 100 users were randomly selected to form 50 groups. The number of requested recommendations (N) was set to 10 in all cases.

Figure 5 shows the results obtained in these experiments. Each bar of the chart at the top represents the average of MAE^G for all the groups of each size and for one elicitation mechanism. The chart at the bottom indicates the average of the distance of the group members' MAE (D^G) of all the groups of each size and for each elicitation mechanism. Both MAE^G and D^G may range between 0 and 1.

As Figure 5 shows, MAE^G of all the mechanisms was quite low (less than 0.06), which demonstrates that the recommendations, in general, are very satisfactory for the groups. In addition, there are no significant differences between the values of MAE^G with respect to the group size, where MAE^G ranged from 0.0226 to 0.057. We found the same situation when we observed the values of D^G. In both cases, as the number of users in the group increases, MAE^G and D^G increase, too.

Comparing the results obtained when applying each technique, in all cases, *Average* was the technique which obtained the worst results with respect to MAE^G. This indicates that the items selected from the group preference model elicited by using this mechanism are not as satisfactory for the group as other mechanisms. Regarding D^G, i.e., the individual satisfaction of each group member with respect to the others, the difference between *Average*, *Average without Misery*, and *II* were not so remarkable.

Average without Misery and *II* obtained the same results in both measures. The reason behind this is that, in most cases, the number of recommendations that the *II* obtains after taking the preferences voted by all the group members is enough to fulfill the group requirements, which is analogous to applying the

[17]We used $\alpha = 70$ for the *AWM* and *II*. In the case of the *ICI* strategy, we used $\delta = 80$ and $\gamma = 95$ in these experiments.

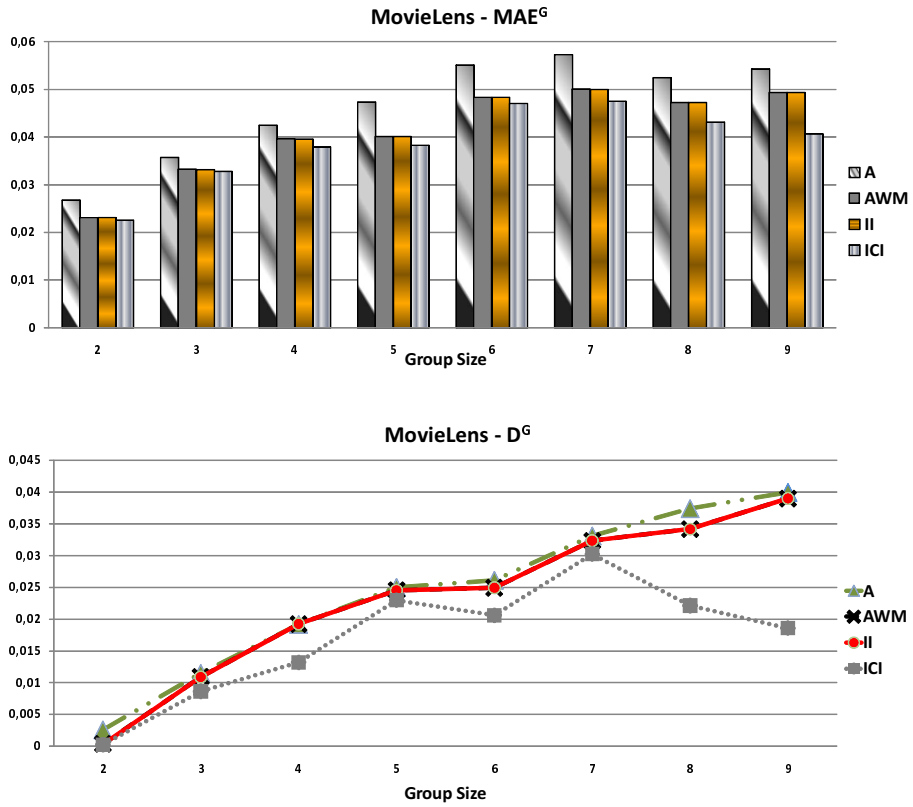

Figure 5: Comparison of all the elicitation mechanisms in the movies domain.

Average without Misery technique. These results turn out to be as expected because this domain has very few features and therefore, it is likely that the *Average without Misery* easily finds features that are common to all members in the group (if there are few features, it is likely that they are shared by the majority of the members). On the other hand, as there are plenty of movies associated to the same feature in the data set, it is likely to find a movie recommendation for a group in almost all situations.

With regard to the *ICI* mechanism, both MAE^G and D^G were lower than the values obtained with the other mechanisms in all cases, which indicates that *ICI* gives better recommendations from the point of view of the whole group and the individual user. This is related to the fact that the *ICI* builds an initial P^G with the preferences that are obtained by the *Average without Misery* whose d^{Gf} is greater than a given δ. This implies that the first list of recommended items (no more iterations are needed) contains those items that match the best-valued preferences for the group, unlike *Average without Misery* and *II*, which consider all the preferences shared by all the group members. In other words,

when the initial P^{AWM} contains many preferences, *ICI* helps to select those that best match the group.

Another important aspect is that all the techniques were able to find the number of requested recommendations (N=10) in all cases due to the small set of features in the ontology and the great volume of data. This facilitates finding movies that are not included in the historical interaction of the users in the group, i.e., movies that have not yet been seen by any user.

6.3. Tourism domain

As explained in section 3.1.2, the tourism domain was developed in our research group. This data set contains information about leisure and tourist activities in the city of Valencia (Spain). The ontology comprises 115 features structured in two levels of the hierarchy (see Figure 3), and the data set stores 158 sites.

Information about the users was collected from 58 real users, who directly filled out a questionaire through a web service that serves as interface with the GRS (see section 3.2.2). Specifically, each user gave personal details and rated the 15 features of the first level in the ontology. Moreover, in order to have data to compare the results obtained by the RS, all the users were requested to rate every item in the data set through a form that was independent from the website. If the users had not visited the place, the rate indicated whether or not they would be interested in visiting it. In this way, we have complete feedback information about the users, unlike in the movies domain, where users only rate the movies they have already seen.

Similarly to the movies domain, all the group preference elicitation techniques[18] were tested with groups of different size ranging from 2 to 9 randomly selected users. Specifically, the whole set of users (58) participated in a group, to have 26 groups of each size. The number of requested recommendations (N) was set to 10 in all cases. We took into account 20% of the total feedback information, meaning that we randomly select around 25 sites visited for each user as his/her historical interaction with the system. Figure 6 shows the results obtained in these experiments. The interpretation of these charts is analogous to the movies domain.

As Figure 6 shows, MAE^G of all the mechanisms is quite variable, ranging from 0.05 to almost 0.4. This indicates that it is more difficult to give accurate recommendations in this domain. With respect to D^G, there are clear differences among the evaluated mechanisms. However, all mechanisms obtained similar results for all the group sizes, which indicates that, in this domain, these techniques give similar individual satisfaction independently from the number of users in the group.

Average is the technique that had the worst results with respect to both group and individual satisfaction (MAE^G and D^G, respectively), in all group

[18]The values of the thresholds for each elicitation technique are the same as in the movies domain.

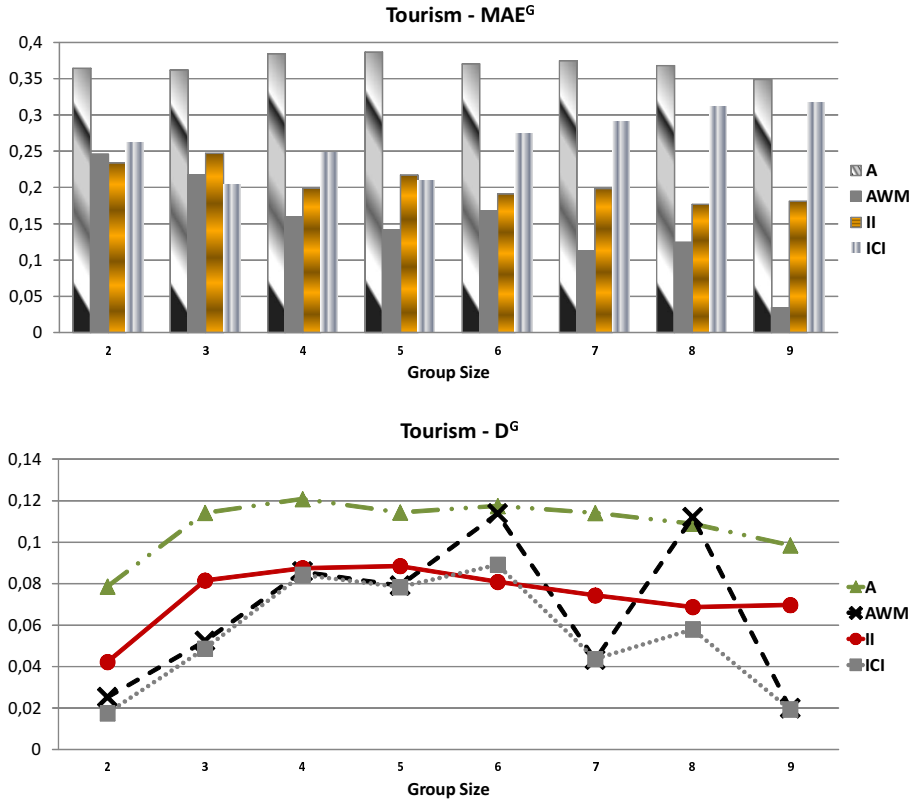

Figure 6: Comparison among all the elicitation mechanisms in the tourism domain.

sizes. However, it was able to return the number of requested recommendations in all cases.

On the other hand, *Average without Misery* was the technique that obtained the lowest MAE^G, in general. However, these values are not always representative because the number of obtained recommendations is quite low compared to other techniques. For example, it returned 7.2 items on average for groups of 2 people, 2.18 for groups of 5 people and 1.17 for groups of 9 people. In contrast, *II* always returned more than 9 items on average. This indicates that *Average without Misery* cannot be used in domains where there is not a wide variety of items to select.

II obtained a low MAE^G, which was always lower than 0.25, but it is obviously higher than the MAE^G of the *Average without Misery* mechanism. However, an important difference between these two mechanisms is that *II* obtained all the required recommendations (N=10) for almost all group sizes, as explained above. With respect to individual satisfaction, *II* obtained a low D^G, which was always lower than 0.09, which indicates that there are no significant differences

	Movies Domain	Tourism Domain
Users' profiles	900	58
Items	1682	158
Features in the ontology	20	115
Average number of rated items by user	45	158
Average number of features in the user GL^u	15	15
Average number of features that describe an item	1.72 (11.63%)	2.55 (2.21%)
Ontology	Flat	Hierarchical
User information	Very little	Enough

Table 7: Comparison of the movies and the tourism domains.

among the individual satisfaction of the group members.

The *ICI* obtained recommendations with a quite high MAE^G, which means that they are not good recommendations for the groups as a whole. The reason behind this is that, as explained in section 5.2.2, the *ICI* algorithm depends on the group preference model elicited by the *Average without Misery* mechanism. From this first P^G, *ICI* completes it in order to obtain a better list of recommendations. However, if the *Average without Misery* mechanism obtains a poor P^G, the *ICI* has more difficulties to obtain good recommendations because the meta-user does not adequately represent the group members. On the contrary, D^G was the lowest in this domain when using this technique. This is because the recommendations are equally *not* satisfactory for all the group members.

A comparison of the mechanisms that are able to return the number of required recommendations shows that *Average* recommends the most prioritized items for *at least one* member in the group but that the *II* considers the preferences that satisfy a larger number of users, which gives the best results with respect to group and individual satisfaction. All of these results lead us to conclude that *II* is the best technique in this domain.

6.4. Discussion

This section is devoted to establishing a comparison between the results obtained with the various techniques when applied to both domains (movies and tourism domains) by also taking into account the particularities of these domains. The differences between the two domains and ontologies (shown in Table 7) have a large impact and influence on the experimental results [65]. Basically, the common observation is that whereas a technique performs very well in one domain, it does not succeed in the other domain.

The values of MAE^G in both domains were quite different (between 0.0226 and 0.057 in the movies domain vs. between 0.05 and 0.4 in the tourism domain). We found the same situation when analyzing D^G. These values indicate that making accurate recommendations in the movies domain is easier than in the tourism domain. The reasons for this are the following. In the first place, the data volume is much greater in the movies domain than in the tourism

domain, so there are more items available for recommendation in the former. Thus, in movies, it is easier to find *new items* that satisfy the group preferences, i.e., items not contained in the historical interaction of the user. Moreover, the average number of features per movie is 1.72 while tourist sites are associated to 2.55 features on average. At first glance, the more features associated to an item, the easier it is to find items that satisfy the group preferences. However, in the movies ontology, the overall number of features is rather small (20) in comparison to the tourism ontology (115), so the percentage of features for defining a movie, considering the total number of features associated to a particular movie, is greater than in the tourism domain (11.63% versus 2.21%).

The number of recommendations obtained on average for each group size in the movies domain was always 10, whereas in the tourism domain, it depended on the technique. This is clearly due to the great number of items in the Movies data set in comparison to the tourism domain.

Remember that an important goal when developing our GRS, where all the users in a group play the same role, is that no member in the group be especially dissatisfied with the decisions. This implies that the satisfaction of all the members is quite similar, i.e., it has a low D^G. Therefore, the best mechanism is the one that brings together a high group satisfaction (low MAE^G) and a high individual satisfaction (low D^G) [39].

In summary, the *Average* operation elicits a group preference model by aggregating the preferences of the users in the group, associating a value to the group preference which is the average of the individual degrees of interest for each preference. The main drawback of this approach is that it may leave some members rather dissatisfied. On the other hand, the *Average without Misery* operation creates a group preference model with the preferences above a threshold that are shared by all the group members. This approach also has the limitation that the result of this aggregation might lead to very few recommended items.

Even though these mechanisms seem quite straightforward, the experiments performed with these techniques show they give fairly good results. However, these techniques get worse depending on the domain characteristics (when the volume of data is reduced or the items are classified with a more complex ontology) or the group characteristics (size, heterogeneity, groups whose members have rated so many items that there are not enough items left to recommend). In this case, the group satisfaction of both mechanisms decreases, but it does so more drastically with the *Average without Misery* mechanism. The user satisfaction decreases also, but it decreases more with the *Average* mechanism. In the situations where these techniques fail to find enough preferences common to all members or the contrary, too many common preferences, more sophisticated techniques like *II* or *ICI* are needed.

II focuses on the items that match a set of preferences that are *shared by most of the members in the group.* Based on a *voting* strategy, it incrementally relaxes the number of users that must satisfy a particular preference. Thus, we will always be able to give recommendations for a group. Specifically, it includes the items obtained using the *Average without Misery* technique in the recommendation list, and it incrementally adds items to the recommendation

list by considering preferences with a lower number of votes. The new items increase the group satisfaction, but as they are only shared by some of the group members, individual satisfaction decreases.

ICI focuses on the items that match a set of preferences *with the highest degree of interest* and the items preferred by other users who liked this first set of items. By using these preferences, the *ICI* obtains new preferences by means of a collaborative RS. It obtains items that greatly increase the group satisfaction in domains like movies. However, it behaves worse in domains like tourism, where the possibilities of finding many users that have similar tastes than the group is much harder.

With regard to the best technique in each domain, *ICI* obtains the best results in the movies domain, whereas *II* reports the best MAE^G and D^G in the tourism domain. This leads us to consider that a hybrid technique would be interesting.

Finally, we analyze the temporal performance of each mechanism in both domains. First, our analysis reveals that the largest amount of processing time is devoted to accessing the database. Therefore, an improvement in this would lead to an important improvement in the temporal performance. Second, the temporal performance strongly depends on the number of users in the group. This is due to the fact that, as the group size increases, it is much harder to find items that satisfy all the group members that do not belong to the users' historical interaction with the system. This means that a greater number of accesses to the database are needed to retrieve the items that fulfill the previous conditions.

In the movies domain, all the mechanisms scale well with groups between 2 and 7 users and there are no significative differences between each technique with respect to the others. However, in groups of 8 and 9 members, the temporal performance gets worse, in general. Specifically, both *Average without Misery* and *II* undergo a steep increase in the temporal cost when comparing the performance of groups of 7 and 8 members, which is not as remarkable in groups of 9 users. On the other hand, the *Average* also scales well in groups of 8 users, but it is the worst mechanism when dealing with groups of 9 members. Finally, the *ICI* scales fairly well with all group sizes, despite applying a collaborative RS. The reason behind this is that the information about the users is preprocessed and loaded into memory so that the access to this data is much faster.

With respect to the tourism domain, the temporal performance is much better than in the movies domain, mainly due to the small size of the database. Therefore, we think that the analysis in the movies domain is more representative of the performance of our GRS.

7. Conclusions and further work

This paper describes a group recommender system that is capable of offering a recommendation for a group of users over a range of different domains.

It builds a group preference model by merging the preference model of the individual users, which results in a set of group preferences that are labeled with a degree of interest. The individuals' preferences are elicited using a hybrid technique that mixes four techniques, namely demographic, collaborative, content-based, and general-likes filtering. The group profile is thus composed of a set of preferences and the GRS elicits the items that best match these preferences.

The main contribution of this paper is the introduction and comparison of four techniques for eliciting the group preference model from the individuals' models. Techniques already applied in other GRSs, such as the *Average* and the *Average without Misery*, report good results, but they get worse depending on the domain characteristics. In situations where these techniques fail to find enough recommended items for all members, other techniques, such as the Incremental Intersection or the Incremental Collaborative Intersection, are introduced to alleviate some drawbacks of the former techniques, as the experiments show.

With regard to further work, we are currently developing a Multi-Agent architecture for GRS [58, 59]. Both users and system components are implemented as agents. The recommendation techniques (both BRTs and techniques for eliciting group preferences) are modelled as agents. This provides flexibility, openness, adaptability, and scalability to our GRS. Given that each member of the group is an agent, groups could change dynamically and a user could come into the group or leave the group during the recommendation process. The recommendation process is dynamically adapted to the new number of group components. Besides these advantages, the most important issue in this architecture is that the user agents make use of agreement techniques to obtain the group recommendations. In this sense, we are following two research lines.

In the first research line, the group members (user agents) negotiate with their individual preferences in order to obtain the group preferences model [18, 55], by means of an alternative offers protocol with a mediator. If an agreement is reached, the negotiated group preferences model is used to obtain the group recommendation. The user agents attempt to achieve a reconciled solution for the whole group maximizing the user satisfaction. These agents can adopt different behaviours during the negotiation process (self-interested, collaborative or highly collaborative). The inclusion of these techniques will allow us to account for more sophisticated user behaviors in the group.

In the second research line, we are working to give the user agents the capability of reasoning about which group preference elicitation technique (*Average*, *Average without Misery*, *Incremental Intersection*, or *Incremental Collaborative Intersection*) is more appropriate to obtain a group recommendation, according to, for example, user likes, user past experience with the techniques, or group heterogeneity. For instance, an argument could defend the hypothesis that the group homogeneity makes the *Average* mechanism more suitable for the group. The user agents would build arguments for or against the use of each technique. The objective is to reach an agreement about the most appropriate technique by using these dialogues.

References

[1] G. Adomavicius, A. Tuzhilin, Personalization technologies: a process-oriented perspective, Communications of the ACM 48 (2005) 83–90.

[2] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (2005) 734–749.

[3] L. Ardissono, A. Goy, G. Petrone, M. Segnan, P. Torasso, INTRIGUE: personalized recommendation of tourist attractions for desktop and handset devices, Applied AI, Special Issue on Artificial Intelligence for Cultural Heritage and Digital Libraries 17 (2003) 687–714.

[4] K. Arrow, Social Choice and Individual Values, Yale University Press, 1963.

[5] M. Balabanovic, Y. Shoham, Fab: Content-based collaborative recommender, Communications of the ACM 40 (1997) 66–72.

[6] A. Barragans, E. Costa, J. Burguillo, M. Rey, F. Mikic, A. Peleteiro, A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition, Information Sciences 180 (2010) 4290–4311.

[7] J. Baskin, S. Krishnamurthi, Preference aggregation in group recommender systems for committee decision-making, in: Proceedings of the ACM conference on Recommender systems RecSys, pp. 337–340.

[8] C. Basu, H. Hirsh, W. Cohen, Recommendation as classification: using social and content-based information in recommendation, in: Proceedings of the National Conference on Artificial Intelligence (AAAI 1998), pp. 714–720.

[9] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: Bagging, boosting, and variants, Machine Learning 36 (1999) 105–139.

[10] Y. Blanco, M. Lopez, J. Pazos, Adapting spreading activation techniques towards a new approach to content-based recommender systems, in: Intelligent Interactive Multimedia Systems and Services, volume 6 of *Smart Innovation, Systems and Technologies*, Springer Berlin Heidelberg, 2010, pp. 1–11.

[11] L. Boratto, S. Carta, State-of-the-art in group recommendation and new approaches for automatic identification of groups, in: A.G. Soro A., Vargiu E., P. G. (Eds.), Information Retrieval and Mining in Distributed Environments, volume 324 of *Studies in Computational Intelligence*, Springer Berlin / Heidelberg, 2011, pp. 1–20.

[12] L. Boratto, S. Carta, M. Satta, Groups identification and individual recommendations in group recommendation algorithms, in: Workshop on the Practical Use of Recommender Systems, Algorithms and Technologies (PRSAT), held in conjunction with RecSys 2010, pp. 27–34.

[13] S. Brams, P. Fishburn, Handbook of social choice and welfare, Handbook of Social Choice and Welfare, Elsevier, 2002, pp. 173–236.

[14] R. Burke, The adaptive web, The Adaptive Web, Springer Berlin / Heidelberg, 2007, pp. 377–408.

[15] L. de Campos, J. Fernandez-Luna, J. Huete, M. Rueda-Morales, Managing uncertainty in group recommending processes, User Modeling and User-Adapted Interaction 19 (2009) 207–242.

[16] E. Elkind, H. Lipmaa, Hybrid voting protocols and hardness of manipulation, in: Proceedings of the International Symposium on Algorithms and Computation (ISAAC 2005), Springer-Verlag, 2005, pp. 206–215.

[17] D. Fesenmaier, F. Ricci, E. Schaumlechner, K. Wober, C. Zanella, Dietorecs: Travel advisory for multiple decision styles, in: Proceedings of the International Conference on Information and Communication Technologies in Tourism 2003, pp. 232–241.

[18] I. Garcia, L. Sebastia, E. Onaindia, A negotiation approach for group recommendation, in: Proceedings of the International Conference on Artificial Intelligence (IC-AI 2009), pp. 919–925.

[19] I. Garcia, L. Sebastia, E. Onaindia, On the design of individual and group recommender systems for tourism, Expert Systems with Applications 38 (2011) 7683–7692.

[20] I. Garcia, L. Sebastia, E. Onaindia, C. Guzman, A group recommender system for tourist activities, in: T. Di Noia, F. Buccafurri (Eds.), Proceedings of the International Conference on Electronic Commerce and Web Technologies (EC-Web 2009), volume 5692 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2009, pp. 26–37.

[21] I. Garcia, L. Sebastia, S. Pajares, E. Onaindia, GRSK: A generalist recommender system, in: Proceedings of the International Conference on Web Information Systems ans Techologies (WEBIST 2010), pp. 211–218.

[22] M. Ghazanfar, A. Prugel-Bennett, A scalable, accurate hybrid recommender system, in: The International Conference on Knowledge Discovery and Data Mining (WKDD 2010), pp. 94–98.

[23] D. Goldberg, D. Nichols, B. Oki, D. Ferry, Using collaborative filtering to weave an information tapestry, Communications of the ACM 35 (1992) 61–70.

[24] F. Goossen, W. IJntema, F. Frasincar, F. Hogenboom, U. Kaymak, News personalization using the cf-idf semantic recommender, in: Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS 2011), pp. 10:1–10:12.

[25] T. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (1993) 199–220.

[26] U. Hanani, B. Shapira, P. Shoval, Information filtering: Overview of issues, research and systems, User Modeling and User-Adapted Interaction 11 (2001) 203–259.

[27] T. Ho, J. Hull, S. Srihari, Decision combination in multiple classifier systems, IEEE Trans. Pattern Analysis and Machine Intelligence 16 (1994) 66–75.

[28] A. Jameson, More than the sum of its members: Challenges for group recommender systems, in: Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI 2004). Gallipoli, Italy, pp. 48–54.

[29] A. Jameson, S. Baldes, T. Kleinbauer, Two methods for enhancing mutual awareness in a group recommender system, in: Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI 2004). Gallipoli, Italy, pp. 447–449.

[30] A. Jameson, B. Smyth, The adaptive web. lncs, The Adaptive Web. LNCS, volume 4321 of *LNCS*, Springer, Heidelberg, 2007, pp. 596–627.

[31] B. Jeong, J. Lee, H. Cho, Improving memory-based collaborative filtering via similarity updating and prediction modulation, Information Sciences 180 (2010) 602–612.

[32] T. Kamishima, H. Kazawa, S. Akaho, A survey and empirical comparison of object ranking methods, in: Preference Learning, Springer Berlin Heidelberg, 2011, pp. 181–201.

[33] H. Kim, J. Kim, Y. Ryu, Personalized recommendation over a customer network for ubiquitous shopping, IEEE Transactions on Services Computing 2 (2009) 140–151.

[34] P. Kourouthanassis, D. Spinellis, G. Roussos, G. Giaglis, Intelligent cokes and diapers: mygrocer ubiquitous computing environment, in: Proceedings of the International Conference on Mobile Business (M-business 2002), pp. 150–172.

[35] S. Lee, Y. Cho, S. Kim, Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations, Information Sciences 180 (2010) 2142–2155.

[36] H. Lieberman, N.V. Dyke, A. Vivacqua, Let's browse: A collaborative browsing agent, Elsevier Science B. V 12 (1999) 427–431.

[37] D. Liu, C. Lai, W. Lee, A hybrid of sequential rules and collaborative filtering for product recommendation, Information Sciences 179 (2009) 3505–3519.

[38] F. Lorenzi, F. Santos, J. Paulo, R. Ferreira, A. Bazzan, Optimizing preferences within groups: A case study on travel recommendation, in: Z. G., da Costa A. (Eds.), Advances in Artificial Intelligence (Brazilian Symposium on Artificial Intelligence, SBIA), volume 5249 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2008, pp. 103–112.

[39] J. Masthoff, Group modeling: Selecting a sequence of television items to suit a group of viewers, User Modeling and User Adapted Interaction 14 (2004) 37–85.

[40] J. Masthoff, Recommender systems handbook, Recommender Systems Handbook, Springer, 2011, pp. 677–702.

[41] J. McCarthy, T. Anagnost, MusicFX: An arbiter of group preferences for computer supported collaborative workouts, in: Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW 1998), pp. 363–372.

[42] K. McCarthy, M. Salam, L. Coyle, L. McGinty, B. Smyth, P. Nixon, CATS: A synchronous approach to collaborative group recommendation, in: Proceedings of the Florida Artificial Intelligence Research Society Conference (FLAIRS 2006), pp. 1–16.

[43] B.N. Miller, I. Albert, S. Lam, J.K. J., Riedl, Movielens unplugged: experiences with an occasionally connected recommender system, in: Proceedings of the International Conference on Intelligent User Interfaces (IUI 2003), pp. 263–266.

[44] C. Musto, Enhanced vector space models for content-based recommender systems, in: Proceedings of the ACM conference on Recommender systems (RecSys 2010), pp. 361–364.

[45] M. O'Connor, D. Cosley, J. Konstan, J. Riedl, PolyLens: a recommender system for groups of users, in: Proceedings of the European Conference on Computer Supported Cooperative Work (ECSCW 2001), pp. 199–218.

[46] M. Pazzani, A framework for collaborative, content-based and demographic filtering, Artificial Intelligence Review 13 (1999) 393–408.

[47] M. Pazzani, D. Billsus, Learning and revising user profiles: The identification of interesting web sites, Machine Learning 27 (1997) 313–331.

[48] M. Pazzani, D. Billsus, The adaptive web, The Adaptive Web, Springer Berlin / Heidelberg, 2007, pp. 325–341.

[49] C. Plua, A. Jameson, Collaborative preference elicitation in a group travel recommender system, in: Proceedings of the AH 2002 Workshop on Recommendation and Personalization in eCommerce. Malaga, Spain., pp. 148–154.

[50] C. Porcel, A. Tejeda-Lorente, M. Martinez, E. Herrera-Viedma, A hybrid recommender system for the selective dissemination of research resources in a technology transfer office, Information Sciences In press (2011).

[51] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, J. Riedl, Grouplens: An open architecture for collaborative filtering of netnews, in: Proceedings of the Computer Supported Collaborative Work Conference (CSCW 1994), pp. 175–186.

[52] P. Resnick, H. Varian, Recommender systems, Communications of the ACM 40 (1997) 56–58.

[53] F. Ricci, B. Arslan, N. Mirzadeh, A. Venturini, ITR: A case-based travel advisory system, in: Proceedings of the European Conference on Advances in Case-Based Reasoning (ECCBR 2002), Lecture Notes In Computer Science, volume 2416, pp. 613–627.

[54] T. Ruotsalo, Methods and applications for ontology-based recommender systems, Ph.D. thesis, Aalto University, School of Science and Technology (Espoo, Finland), 2010.

[55] M. Salamo, K. McCarthy, B. Smyth, Generating recommendations for consensus negotiation in group personalization services, Personal and Ubiquitous Computing (2011) 1–14.

[56] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the International Conference on World Wide Web (WWW 2001), pp. 285–295.

[57] J. Schafer, J. Konstan, J. Riedl, Recommender systems in e-commerce, in: Proceedings of the ACM Conference on Electronic Commerce (EC 1999), ACM, 1999, pp. 158–166.

[58] L. Sebastia, I. Garcia, A. Giret, A multiagent architecture for tourism recommendation, in: Springer (Ed.), Proceedings of the International Conference on Practical Applications of Agents and Multiagents Systems, Workshop on Artificial Intelligence and Distributed Systems (PAAMS-AIDS 2010), volume 2, pp. 547–554.

[59] L. Sebastia, I. Garcia, A. Giret, A multiagent architecture for single user and group recommendation in the tourism domain, International Journal of Artificial Intelligence (IJAI). To appear (2011).

[60] L. Sebastia, I. Garcia, E. Onaindia, C. Guzman, e-Tourism: a tourist recommendation and planning application, International Journal on Artificial Intelligence Tools (WSPC-IJAIT) 18 (2009) 717–738.

[61] G. Semeraro, P. Lops, P. Basile, M. de Gemmis, Knowledge infusion into content-based recommender systems, in: Proceedings of the ACM conference on Recommender systems (RecSys 2009), pp. 301–304.

[62] M. van Setten, S. Pokraev, J. Koolwaaij, Context-aware recommendations in the mobile tourist application COMPASS, in: P. De Bra, W. Nejdl (Eds.), Adaptive Hypermedia and Adaptive Web-Based Systems, volume 3137 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2004, pp. 515–548.

[63] M. van Setten, J. Reitsma, P. Ebben, Duine toolkit - user manual, Technical Report, Telematica Instituut, 2006.

[64] P. Symeonidis, A. Nanopoulos, A. Papadopoulos, Y. Manolopoulos, Collaborative recommender systems: combining effectiveness and efficiency, Expert Systems with Applications 34 (2008) 2995–3013.

[65] L. Tao, S. Anand, Exploiting domain knowledge by automated taxonomy generation in recommender systems, in: Proceedings of the International Conference on Electronic Commerce and Web Technologies (EC-Web 2009), T. Di Noia and F. Buccafurri eds, LNCS 5692, Springer-Verlag, 2009, pp. 120–131.

[66] B. Towle, C. Quinn, Knowledge Based Recommender Systems Using Explicit User Models, Technical Report, Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, AAAI Technical Report WS-00-04, 2000.

[67] M. Vozalis, K. Margaritis, Using svd and demographic data for the enhancement of generalized collaborative filtering, Information Sciences 177 (2007) 3017–3037.