Recommendation Based Discovery of Dynamic Virtual Communities

Manos Papagelis[1, 2] and Dimitris Plexousakis[1, 2]

[1] Institute of Computer Science, Foundation for Research and Technology - Hellas
P.O. Box 1385, GR-71110, Heraklion, Greece
{papaggel, dp}@ics.forth.gr
[2] Department of Computer Science, University of Crete
P.O. Box 2208, GR-71409, Heraklion, Greece

Abstract. Recommendation systems are becoming increasingly popular in various large-scale web-based applications (such as infomediaries, e-marketplaces, knowledge portals) since they enable users and customers to better fulfil their information or product search needs according to their interests. Techniques used to form the recommendations include algorithms from the area of machine learning and vote theory. We introduce an alternative way of building user profiles based on explicit and implicit ratings as well as profile-based recommendation and user-matching algorithms. The paper advocates the use of such algorithms as a tool to automate the discovery and creation of dynamic, virtual communities. Furthermore, rating prediction algorithms are proposed as a means of enhancing the interaction between users and a recommendation system. As an application of the proposed techniques, we present MRS, a web-based information system that provides personalized recommendations for cinema movies.

1 Introduction

Recommendation systems [21, 23, 25] has been a popular topic of research ever since the ubiquity of the web made it clear that people of hugely varying backgrounds would be able to access and query the same underlying data. The initial human computer interaction challenge has been made even more challenging by the observation that customized services require sophisticated data structures and well thought-out architectures to be able to scale up to thousands of users and beyond.

In recent years, such systems are extensively adopted by both research and e-commerce applications in order to provide an intelligent mechanism to filter out the excess of information available and to provide customers with the prospect to effortlessly find out items that they will probably like according to their logged history of previous purchases and transactions. Acknowledged "e-markets" that take advantage of such techniques are Amazon.com (www.amazon.com), CDNOW (www.cdnow.com), Drugstore.com (www.drugstore.com), eBay.com (www.ebay.com), Alexa (www.alexa.com), Myfreddy (www.myfreddy.com) and Reel.com (www.reel.com). Numerous non-profitable examples of recommendation systems exist as well such as GroupLens [6], NewsWeeder [16] and MovieLens (www.movielens.umn.edu).

In this paper we introduce an alternative way of building user profiles [11,13,19] by exploiting user ratings in both an explicit and an implicit way [4,5,9,13]. Explicit ratings given to an item are also given implicitly to all the categories that this item belongs to. Taking advantage of this we build a user profile that is based on user preference [19] to specific categories. This alternative way of constructing user profiles permits the development of novel recommendation algorithms. Furthermore, the paper sternly implies the use of these algorithms as a tool to automate the discovery of dynamic virtual communities [17]. The "Community Coherence Grade" term is introduced in order to describe the extent in which the members of a community are related and helps out to discover the looseness or tightness of formed communities. Rating prediction algorithms [26] are also described and the way that they can be incorporated to such systems is discussed so as to provide a kind of web intelligence [27] and to enhance the interaction [14, 29] between users and system. On top of that theoretical approach a web-based application has been developed to form recommendations for cinema movies.

The rest of the paper is organized as follows. Section 2 describes the existing approaches for building user profiles and the existing recommendation methods. Section 3 presents our proposal for user-matching and recommendation algorithms and section 4 describes the way in which these algorithms are used to form communities. Section 5 is dedicated to the description of the implemented system, MRS, and the challenges of porting such algorithms to web are discussed. Section 6 summarizes our contribution and draws directions on further research.

2 Recommendation Systems and User Profiling

2.1 Background

There are generally two methods to form recommendations both depending on the type of items to be recommended as well as on the way that user profiles are constructed and manipulated. The two different approaches are content-based [11,13,27] and collaborative filtering [2,3,18,22,24], while additional hybrid techniques have been proposed as well [2,3,19].

Content-based algorithms are principally used when documents are to be recommended. Such items can be web pages, publications, jokes, news and descriptions of pictures. The system maintains information about user preferences either by initial user input about his interests during the registration process or by rating documents. Recommendations are formed by taking into account the content of documents and by filtering out the ones that better match with user's preferences and built profile.

On the other hand, systems adopting the collaborative-filtering approach, aim to identify users that have relevant interests and preferences with a particular user. This approach applies user-matching techniques in order to discover and exploit similarities and dissimilarities between user profiles. The idea behind this method is

that, it may be of benefit to one's search for information to consult the behavior of other users who share the same or relevant interests and whose opinion can be trusted.

2.2 User Profiling

Under the assumption that items of a specific area of interest are concerned (e.g. movies, CDs, books) it is convenient to consider categorization of the items. Hence, every item belongs to at least one category (e.g. Comedy, Crime, Drama for movies, Pop, Rock, Jazz for CDs). When users rate items, ratings are considered as explicit ratings to the specific item, but also as implicit ratings [4,5,9,13] to the categories that the specific item belongs to. These alternative ways to consider ratings are used to construct two alternative user models respectively and are described below.

2.3 User Model Based on Explicit Ratings

Explicit ratings are considered when a specific user rates a specific item with a specific rating. This information is logged as shown in Table 1 with user's model comprises the total of the explicit ratings that the user has given. The more ratings a user submits, the richer the user's model becomes. The constructed model is very helpful when trying to match users by collaborative filtering algorithms and vector-based similarity calculations, when trying to predict the vote that another user will give to the same movie and when trying to form personalized recommendations [20].

Table 1. User's ratings to a specific Item

UserID	ItemID	Rating
User1	446	5
User2	59	10

2.4 User Model Based on Implicit Ratings

Explicit ratings to an item are also given implicitly to the categories that this item belongs to. The user model built by implicit ratings expresses user's likes or dislikes for specific categories. Taking for example a movie title that belongs to categories Crime and Drama, if a user gives a good rating (bigger or equal to 6) to that movie then the attributes of his profile that represent liking in these categories are augmented (CrimePos, DramaPos); Otherwise if he gives a bad rating (smaller or equal to 5) to the same movie then the attributes of his profile that represent non-liking in these categories are augmented (CrimeNeg, DramaNeg) (Table 2).

Table 2. Snapshot of user profile according to category preferences

UserID	CrimePos	CrimeNeg	DramaPos	DramaNeg
User1	23	17	5	1
User2	12	13	7	2

3 "Find Buddy" and Recommendation Algorithms

3.1 Introduction

Two methods of finding buddies will be described. The first is applied on the user model "based on explicit ratings" and the second on the user model "based on implicit ratings". The first will also be called as "user model according to explicit ratings", and the second as "user model according to user preference to categories". Before describing these methods we briefly present the Pearson Correlation Coefficient that is employed by both of them.

3.2 Pearson Correlation Coefficient

User-matching methods use the Pearson Correlation Coefficient (PCC) [30] in order to calculate the relevance between two models. The PCC is a measure of linear association between 2 vectors (variables). It is useful when attempting to determine if there is a significant relationship between these two vectors (variables). Values of the correlation coefficient range from -1 to +1. The absolute value of the correlation coefficient indicates the strength of the linear relationship between the variables, with larger absolute values indicating stronger relationships. The sign of the coefficient indicates the direction of the relationship. The Pearson Correlation Coefficient is described by the equation below (Eq.1) where X, Y represent the rating vectors of users x, y which refer to the same set of items, Uxi, Uyi represent the rating of user x, y to the i-th item of the set and Uavg_x, Uavg_y represent the rating average of users x, y based on this set of items.

$$PCC(X,Y) = \frac{\sum_{1}^{k}(U_{xi} - U_{avg\_x})(U_{yi} - U_{avg\_y})}{\sqrt{\sum_{1}^{k}(U_{xi} - U_{avg\_x})^2 . \sum_{1}^{k}(U_{yi} - U_{avg\_y})^2}}, i = 1,2,...,k \quad (1)$$

In order to use the PCC we build vectors for each model representing the users' behavior concerning the same set of data. As far as the first method is concerned vectors consist of the values of the explicit ratings given to the same set of items by the two users (Fig.1, left part). As far as the second method is concerned, vectors consist of the attributes of the user's profile that express preference (e.g. CatA_Pos) (Fig.1, right part).

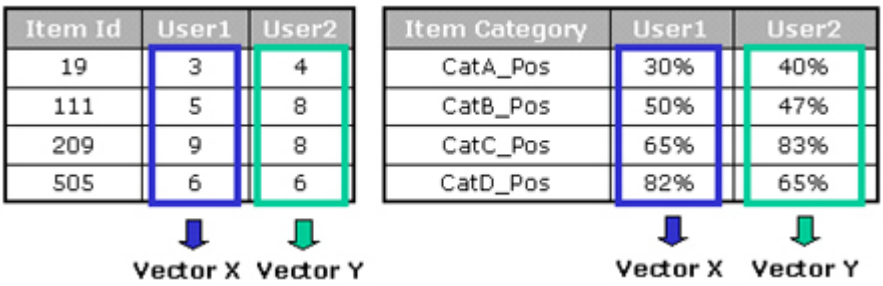

Fig.1. Vectors that are constructed by the two alternative user models

3.3 Find Buddies Based on the Model Built According to Ratings

The following algorithm is used in order to find "buddies" according to explicit ratings given to specific items. In order to use the Pearson Correlation Coefficient we must compare the same set of data for two users. This is satisfied by selecting only items that both of users have rated. The algorithm is going to calculate the relevance of the profiles and will decide whether the subject users are relative or not, that is whether they have similar rating activity as far as the common set of movies is concerned (Algorithm 1).

1. Find users that already have rated at least once
2. For each one of them compare its profile, built according to the explicit ratings to specific items, with the profile of the subject user
    a. Find the items that both have rated
    b. Use Pearson Correlation Coefficient to calculate the relevance between the two profiles
3. If the correlation coefficient is greater than a quality threshold then consider the user as "buddy" else not
4. Continue with the second step until there are not remaining users

Algorithm 1. Find Buddies according to explicit ratings

The complexity of the algorithm is linear on the number of users on the system that have rated at least once.

3.4 Find Buddies Based on Model Built According to User Preference for Categories

As for each user a profile based on implicit ratings to categories has been developed it would be valuable to have an algorithm that can compare this profile to other user profiles. The following algorithm is proposed in order to apply the "find buddy" algorithm to the user model built according to user preference for categories. The

algorithm calculates the relevance of the profiles and decides whether the subject users are relative or not, that is whether they have similar preferences for specific categories of items (Algorithm 2).

1. Find users that have rated at least once (i.e., already have profile)
2. Find the relevance between each user and the subject user
    a. Take the occasional user's profile
    b. Take the subject user's profile
    c. Calculate the Pearson Correlation Coefficient between the two profiles
3. If the correlation coefficient is greater than a quality threshold then consider the user as "buddy" else not
4. Continue with the second step until there are not remaining users

Algorithm 2. Find Buddies According to user category preferences

The complexity of the algorithm is linear on the number of users on the system that have rated at least once.

3.5 Recommendation Algorithm Based on Explicit User Ratings for Items

In order to form recommendations for a user this algorithm uses the "find buddies based on model built according to ratings" algorithm to find users with similar behavior. We call the set of users found as "consultants", because they will play the role of consultants to the subject user by judging whether an item is worth to be recommended or not. The following algorithm puts into practice the pure collaborative filtering method.

For each item that the user has not rated yet we select a subset of users, which come from the consultants set calculated before, that have already rated the item. In sequence, the average of the consultant ratings to this item is calculated and if it is greater than a quality threshold[1] then it is recommended to the subject user else not. After the termination of the algorithm a set of items will be recommended with priority to items that were assigned the largest average among the consultants (Algorithm 3).

1. Find the consultants of the initial user (i.e., find the buddies of the initial user)
2. Find the items that user has not rated yet
3. For each one of the items find the subset of consultants that have rated this item
4. Calculate the average of the ratings that this subset of consultants has given to the item
5. If the calculated average is larger than a quality threshold then recommend

[1] Quality threshold is a number ranging between 1 and 10 that denotes the bound over which a criterion is considered to be satisfied.

6. Continue with the third step until there are no remaining items

Algorithm 3. Recommendation algorithm based on user explicit ratings to items

Worst case: (max-number of consultants) * (max-number of not rated items)

3.6 Recommendation Algorithm Based on User Preference for Specific Categories

As the user interacts with the system by rating items his profile based on user preference to categories is continuously enriched. We propose the following algorithm in order to form recommendations exploiting that enriched profile.

For each item that the user has not rated yet the system will find out the categories that this item belongs to. In sequence, the system will calculate the percentage of the positive ratings against the percentage of the negative ratings that the user has given to each one of the categories that this item belongs to and finally will calculate the average of the positive percentages. If the average is greater than a quality threshold then the item will be recommended else not (Algorithm 4).

The algorithm steps are shown in details below:

1. Find the items that user has not rated yet
2. For each one of them find the categories in which the item belongs to
3. For each one of these categories calculate the percentage of the positive ratings that the user has given against the negative ones
4. Calculate the average of positive percentages of all these categories
5. If the average is bigger than a quality threshold then recommend it to the user
6. Continue with the second step until there are not any remaining items

Algorithm 4. Recommendation algorithm based on user category preferences

Worst case: (max-number of categories for an item) * (max-number of not rated items)

The equation below (Eq.2) describes the calculation performed by the algorithm for deciding whether a movie is recommended, where k is the number of the categories that an item belongs to, Pos_i is the number of positive ratings that user has given to category i and Neg_i is the number of negative ratings that user has given to category i.

$$Avg_{Movie} = \frac{\sum_{1}^{k} \frac{Pos_i}{Pos_i + Neg_i}}{k}, i = 1, 2, ..., k \quad (2)$$

Collaborative filtering method and user matching techniques can be used in order to provide a kind of web intelligence to a system by introducing interaction sequences

between user and system such as the rating prediction algorithm described below does.

3.7 Rating Prediction Algorithm

The rating prediction algorithm aims to predict the rating that a user is likely to give to an item. The algorithm exploits the logged history of ratings that a user has given and the ratings that other users have given to this item in order to calculate the predicted rating. This is mostly an interactive game [27] between user and system and provides a form of web intelligence to such a system.

In brief, the algorithm calculates the average of ratings already given by the user and adds to this the weighted average of the ratings that the specific item has already be given by other users, considering as weights the relevance of each user to the subject user (Algorithm 5).

1. Find the average of the user's already given ratings
2. Find the users that have rated the specific item
3. For each user calculate the Pearson Correlation Coefficient between the current user and the initial user (This coefficient, which is a weight, determines how this user might influence the result).
4. For each user calculate the average of the ratings that he has given
5. For each user calculate the rating that he has given to the specific item
6. Calculate the weighted average of the users' ratings to the item

The predicted rating is the sum of the initial user's average calculated at the first step and the adjustment calculated at the sixth step of the algorithm

Algorithm 5. Vote Prediction Algorithm

The complexity of the algorithm is linear on the number of users on the system that have rated the specific item.

The equation that calculates the rating that user i will give to item j is shown below where, k is the number of users that have rated the item j, U_i is the average of the initial user's ratings, w_{mi} is the Pearson correlation coefficient between user m and user i, u_{mj} is the rating that user m has given to item j and U_m is the average of the m-th user's ratings (Eq.3).

$$\hat{u}_{ij} = \overline{U}_i + \frac{\sum_{1}^{k} w_{mi}(u_{mj} - \overline{U}_m)}{\sum_{1}^{k} |w_{mi}|}, m = 1, 2, \ldots, k \quad (3)$$

4 Formation of Communities

4.1 Introduction

The recommendation methods described earlier are detecting similarities between users. As described before users draw on these algorithms in order to find buddies, i.e., users with similar interests. In this respect, recommendation algorithms can be used in the formation of communities [17]. Users that have something in common can be viewed as members of virtual communities.

The interesting part of our approach is that the formation of the communities is performed automatically. Users do not have to subscribe in order to participate in a community, not even to specifically express preferences. The communities are virtually, transparently and dynamically created and are continuously transformed as users rate items, improving their profile. It is significant that a user can at the same time be member of many different communities, while according to his point of view he is the initiator of the community formed around him.

Some of these communities consist of users that are quite related while others consist of users that are less related. In order to have a reference to this characteristic of each community the term "Community Coherence Grade" (CCG) is introduced in order to define the tightness or looseness of each community. The CCG, which is described by the equation 4, expresses the average correlation between the subject user i and the other members of the same community.

$$CCG_i = \frac{\sum_{1}^{k} PCC_{im}}{k}, m = 1, 2, ..., k \tag{4}$$

Here k is the number of members of the specific community. Members are the users for whom PCC between them and the subject user is larger than a quality threshold. PCC_{im} is the calculated Pearson Correlation Coefficient of the m-th user and the subject user i.

Communities created in this way are said to be virtual and dynamic. They are considered as virtual because they do exist just for a period of time and can be viewed just from the current user's perspective around which a group of buddies is concentrated (user's vicinity). They are considered as dynamic because any user's single rating can result in a change of the communities that this user belongs to and an adjustment of the CCG of these communities as well. The continued activity of users that interact with the system by rating more and more items brings about a confused behavior of communities in the system. In the following paragraph a filtering technique that facilitates the detection of sub-communities is described.

4.2 Detecting Sub-communities

The algorithm described below provides the possibility to apply filters that concern the degree of preference to specific categories and to detect which of the users satisfy the filtering criteria that have been specified. This algorithm permits the detection of users that are relevant according to restricted criteria. In this manner, this algorithm yields the opportunity to reveal sub-communities of users, communities that are formed not by taking into account the whole of a users profile but just a part of it.

For example we can apply a filter that will find all the users that like items that belong to category_A, don't like items that belong to category_B and are indifferent to the other categories that an item may belong to. The preference can be expressed in a five-level scale. (0-20%, 20-40%, 40-60%, 60-80%, 80-100% denotes highest likeness). So the earlier example can be interpreted as the need to find users that like category_A to a percentage 80-100% and category_B to a percentage 0-20%. The filters can be more complex by combining degree of preference for all categories that an item may belong, expressing the preference according to the five-level scale. In all cases the algorithm returns the users that satisfy all the criteria by applying a Boolean AND operation to different criteria specified by the filtering procedure (Algorithm 6).

1. Find the categories for which criteria have been specified
2. For each category find the percentages that have been specified (one of the five levels described above)
3. Find the users of the system that have rated at least once
4. For each user decide whether he satisfies the criteria
    a. For each category that has been specified calculate the positive percentage that corresponds to the user.
    b. If the percentage satisfies the criteria that have been applied then continue with the next category.
    c. If a user satisfies the criteria of all categories then that user is returned by the algorithm
5. Continue with the fourth step until there are no remaining users

Algorithm 6. Applying filtering criteria to user preferences

The complexity of the algorithm depends on the number of categories on which criteria have been specified and the number of users on the system.

The rest of algorithms described on the paper considered the whole of the user profiles. Some times these methods result in incorrect conclusions, as they are proved incapable to detect users that are partially very relative, although they seem not to be relative if their profile on whole is considered. The algorithm described above can be used in order to discover tight correlations between users according to parts only of their profiles.

5 Case Study – The Movie Recommendation System

5.1 Introduction

We describe a multimedia, model-based system that demonstrates many of the technical challenges and implications arising from building such systems on the web. The particular system is a movie recommendation system that adopts several key ideas spanning from site design [14, 29] to user walk-through and user-input analysis, to "intelligently" provide user recommendations and, consequently, improve user experience. We build on concepts from collaborative filtering, model matching, vector similarity [7,8,9], vote theory (explicit and implicit ratings), clustering (online-communities, user matching), friendly user interfaces [14, 29] (easy navigation and personalized services), as well as on several ideas for efficient and timely data manipulation [8].

5.2 System Overview

We have built MRS around a small set of database driven web pages. These pages communicate with the main algorithm library as well as with the visualization library to produce the end user functionality. In this section we will give a brief description of the user interface. The interested reader is strongly encouraged to visit the web site (http://diat11.csd.uoc.gr) to obtain a more detailed view. We will also outline some key architectural issues that we have dealt with throughout the development of our system. In order to address portability, reliability and scalability issues, a modular architecture approach has been adopted keeping the functionality independent of the repository used and of the area of interest concerned.

5.3 Walk Through

Information about movies stored on the MRS originates from the Internet Movie Data Base site (www.imdb.com). Movies are categorized and are assigned the IMDB rating which comes up by IMDB users. We use IMDB for accuracy, updating and maintenance reasons. In MRS users are separated in two main categories, unregistered and registered. Registered users are recognized by the system through a secure login procedure (Fig.2).

Fig.2. Login procedure.

Ratings coming from registered users weight more than ratings coming from unregistered users. That seems reasonable, as registered users are restricted to one rating per movie for validity. An overview of the MRS website is shown in figure 3.

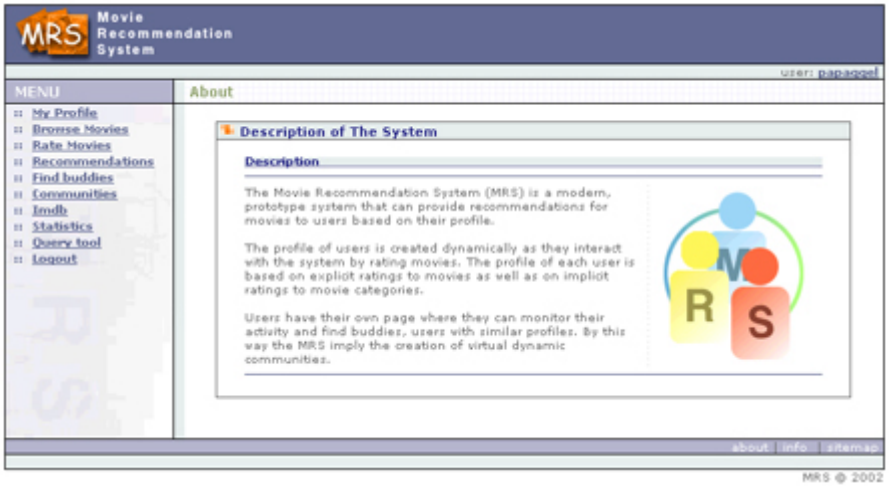

Fig.3. MRS Website Overview

As users interact with the system their profile is continuously improved. The system provides personalized services such as recommendations, rating history and graphical representation of given ratings (Fig.4).

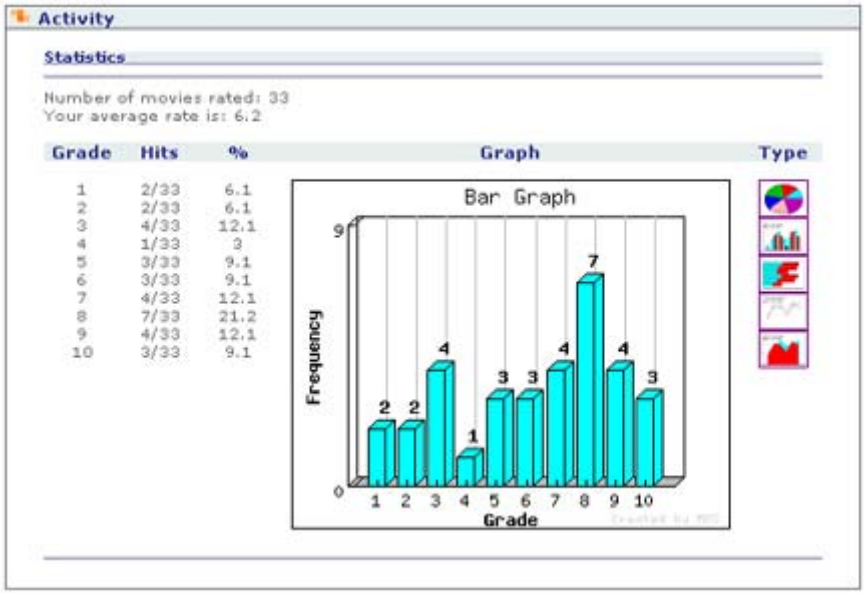

Fig.4. Votes Allocation

The users can browse movies by genre, by release year, alphabetically or by search keyword. Throughout browsing users are encouraged to rate movies that they have already seen. In order to ameliorate the interaction between user and system the rating prediction option is included (Fig.5).

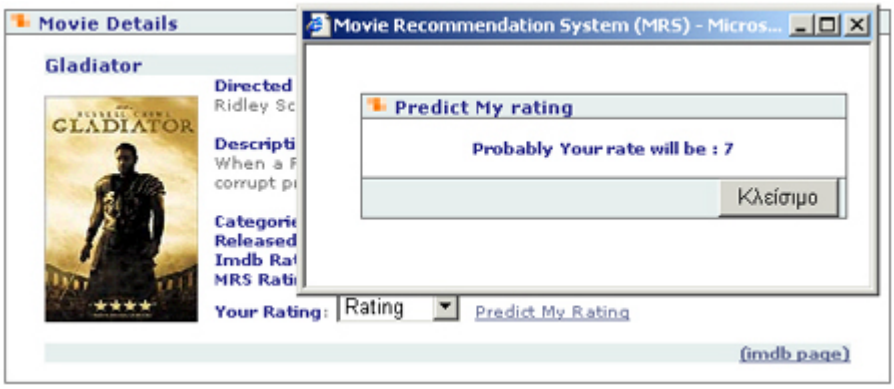

Fig.5. Rating Prediction

The system examines the user's profile and responds to user's request with the rating that he is likely to give to the movie. The accuracy of the algorithm largely depends on the user's earlier activity. If his profile is not poor [28] then the system's reply can be precise.

The "Find Buddy" algorithms described earlier are incorporated into the system under the "Find buddies" selection. The user is able to come across other users with similar interests according to their already developed profile by applying matching techniques. Since there are two ways to maintain user profiles, one according to explicit ratings to movies and another according to genre preferences (explicit ratings used implicitly as ratings to movie categories), there are consequently two implemented methods to find buddies (Fig.6).

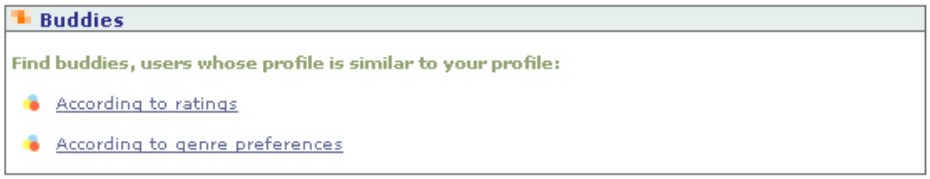

Fig.6. Find Buddies methods

The system acts in response to user request with a series of users that are considered relative according to the algorithm applied each time (Fig.7).

Fig.7. Find buddies method results

This functionality gives the prospect to users to expand relationships and to distribute opinions and thoughts about movies. Since they share common interests they can be considered as a community.

Since the communities brought in the MRS are considered to be virtual and dynamic the need of examining what is going on in the communities comes up. Some tools are described below that have been developed to facilitate monitoring the community activity. The administrator of the site can apply the "Find Buddy" algorithms to specific users. This results in a set of calculated correlations between the specific user and the rest users of the system. Users that satisfy a correlation threshold are considered members of the same community (Fig.8). The average of these correlations represents the "Community Coherence Grade" (CCG) as it was thoroughly explained before.

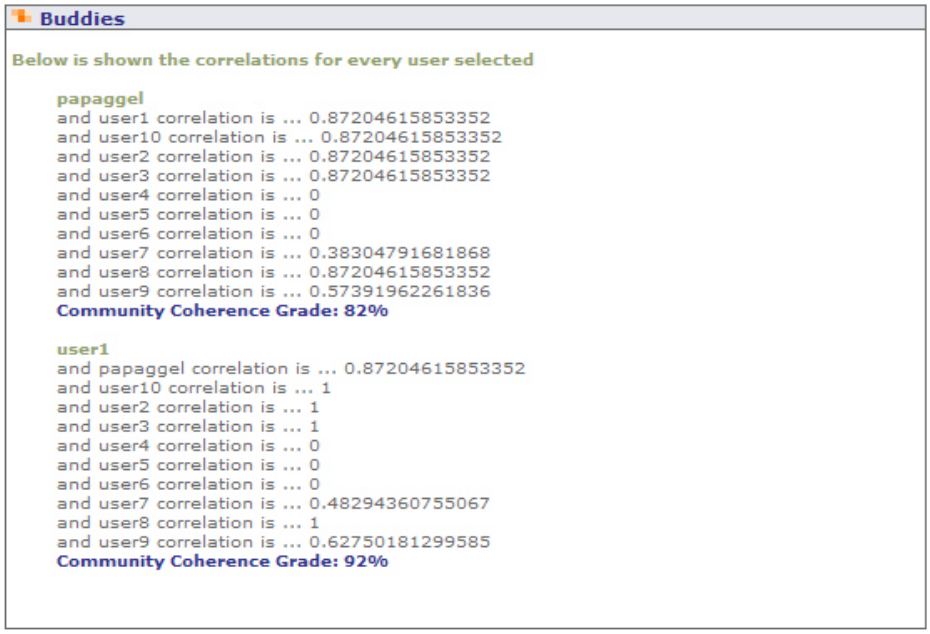

Fig.8. Community Coherence Grade

The CCG measure can be efficiently used to describe the looseness or tightness of a formed community. The greater the CCG of a community is, the more relative the members of the community are considered.

The recommendation algorithms illustrated earlier are integrated into the system in order to provide suggestions for movies to users. The system supports three different methods to form recommendations. The first one does not presuppose the existence of user profiles and, therefore, is not of research interest. The second and the third methods make use of the collaborative filtering algorithms to match similar user profiles. The second method is applied to user profiles that are based on explicit ratings and the third method is applied to user profiles that are based on genre preferences (Fig.9).

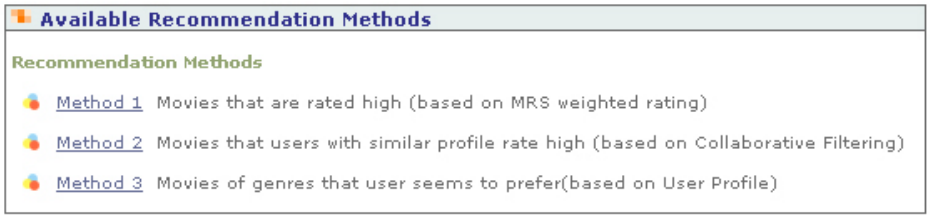

Fig.9. Methods to produce recommendations

In all three cases the system replies with a set of recommended movies that are sorted by level of relevance according to the method that has been selected. An example of the system's response is shown below. The green bar at the upper right of each movie depicts of the level of relevance on a 1 to 10 scale (Fig.10) [14, 29].

Fig.10. Recommendation method result

6 Conclusions and Extensions

The vast volume of information flowing on the web has given rise to the need for information filtering techniques. Recommendation systems are effectively used to filter out excess information and to provide personalized services to users. We described an alternative way to build user model based on user preferences to items that belong to specific categories. Based on this model we discussed new user matching and recommendation algorithms. We advocated the use of these algorithms in order to effectively and efficiently discover communities. These communities are transparently created and they are virtual and dynamic. The "Community Coherence Grade" measure was introduced in order to describe the looseness or the tightness of each community; in other worlds to describe how relative are the community members. The Movie Recommendation System (MRS) was also presented. MRS incorporates well the algorithms described in this paper in order to provide users with personalized recommendations for cinema movies. Future research will focus on qualitative results of the developed algorithms, employment of hybrid algorithms and performance issues.

References

1. D. H. Widyantoro, T. R. Ioerger, and J. Yen. "Learning User Interest Dynamics with a Three-Descriptor Representation". Journal of the American Society for Information Science (JASIS), 2000.
2. M. Balabanovic and Y. Sholam. "Combining Content-Based and Collaborative Recommendation". Communications of the ACM, 40 (3), 1997.
3. G. Karypis. "Evaluation of Item-Based Top-N Recommendation Algorithms", 2000. In Proceedings of CIKM, 2001, pp. 247-254, 2001.
4. David Nichols, "Implicit Rating and Filtering". In Proceedings of Fifth DELOS Workshop on Filtering and Collaborative Filtering, pages 31-36, Budapest, November 1997. ERCIM Report ERCIM-98-W001. Le Chesnay Cedex, France, European Research Consortium for Informatics and Mathematics.
5. Mark Claypool, Phong Le, Makoto Waseda and David Brown, "Implicit Interest Indicators". In Proceedings of the International Conference on Intelligent User Interfaces, Santa Fe, 33-40 2001.
6. P. Resnick, N. Iacovou, M. Suchak, P. Bergston, and J. Riedl. "Grouplens: An open architecture for collaborative filtering of netnews". In Proceedings of CSCW, 1994.
7. R. Baeza-Yates, B. Ribeiro-Neto. "Modern Information Retrieval". Addison Wesley, 1999.
8. T. Mitchell "Machine Learning". NY:McGraw-Hill, 1997.
9. Patrick Baudisch, "Dynamic Information Filtering". Ph.D. Thesis. GMD Research Series 2001, No. 16.
10. M. Pazzani and D. Billsus. "Learning and revising user profiles: The identification of interesting web sites". Machine Learning, 27:313-331, 1997.

11. Abbattista F., Degemmis M., Fanizzi N., Licchelli O., Lops P., Semeraro G., and Zambetta, F. “Learning User Profiles for Content-Based Filtering in e-Commerce”. Italian Artificial Intelligence Conference (AIIA 2002).
12. Wee Sun Lee, “Collaborative Learning for Recommender Systems”. In Proceedings of 18th International Conference on Machine Learning 2001.
13. P. Chan. “A non-invasive learning approach to building web user profiles”. In Proceedings of ACM SIGKDD International Conference, 1999.
14. Daniel Billsus, Clifford A. Brunk, Craig Evans, Brian Gladish and Michael Pazzani, “Adaptive Interfaces for Ubiquitous Web Access”. Communications of the ACM, 45 (5), 2002.
15. M.Balabanovic. “An Adaptive Web Page Recommendation Service”. In Proceedings of First International Conference on Autonomous Agents, pp. 378-385, 1997.
16. Ken Lang, “NewsWeeder: Learning to Filter Netnews”. In Proceedings of the 12th International Conference on Machine Learning 1995.
17. Stuart E. Middleton, Harith Alani, Nigel R. Shadbolt, David C. De Roure, “Exploiting Synergy Between Ontologies and Recommender Systems”. In: The Eleventh International World Wide Web Conference (WWW2002), Hawaii, USA, ACM, Semantic Web Workshop 2002, WWW2002 (2002) Semantic Web Workshop 2002 At the Eleventh International World Wide Web Conference Hawaii, USA.
18. Nick Papadopoulos and Dimitris Plexousakis, “The Role of Semantic Relevance in Dynamic User Community Management and the Formulation of Recommendations”. In proceedings of 14th International Conference on Advanced Information Systems Engineering, CAiSE 2002, Toronto.
19. Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M.McNee, Joseph A. Konstan, John Riedl, “Getting to Know You: Learning New User Preferences in Recommender Systems”. 2002 International Conference on Intelligent User Interfaces.
20. Upendra Shardanand, “Social Information Filtering: Algorithms for Automating "Word of Mouth”. In Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems 1995.
21. “Who’s Who in Recommender Systems”. 2001 ACM SIGIR Workshop on Recommender Systems.
22. Arnd Kohrs and Bernard Merialdo, “Using Category-Based Collaborative Filtering In The Active Webmuseum”. 2000 IEEE International Conference on Multimedia and Expo.
23. J. Ben Schafer, Joseph A. Konstan, and John Riedl, “E-Commerce Recommendation Applications”. Data Mining and Knowledge Discovery 5(1/2): 115-153 (2001).
24. Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl, “Explaining Collaborative Filtering Recommendations”. In proceedings of ACM 2000 Conference on Computer Supported Cooperative Work.
25. Badrul M. Sarwar, George Karypis, Joseph A. Konstan, John T. Riedl, “Application of Dimensionality Reduction in Recommender System -- A Case Study”. In ACM WebKDD Web Mining for E-Commerce Workshop, 2000.

26. Joaquin Delgado, Naohiro Ishii, “Memory-Based Weighted-Majority Prediction For Recommender Systems”. ACM SIGIR'99 Workshop on Recommender Systems, 1999.
27. Dimitrios Kalles, Athanasios Papagelis, Christos Zaroliagis – “A Content-based Web Intelligent System”. IEEE Multimedia 2002.
28. Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, “Methods and Metrics for Cold-Start Recommendations”. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002).
29. Kirsten Swearingen & Rashmi Sinha, “Beyond Algorithms: An HCI Perspective on Recommender Systems”. In proceedings of 2001 ACM SIGIR Workshop on Recommender Systems.
30. Karl Pearson, “Mathematical contribution to the theory of evolution: VII, on the correlation of characters not quantitatively measurable”. Phil. Trans. R. Soc. Lond. A, 195,1-47, 1900.