

# Recommendation Based Discovery of Dynamic Virtual Communities

Manos Papagelis<sup>1,2</sup> and Dimitris Plexousakis<sup>1,2</sup>

<sup>1</sup> Institute of Computer Science, Foundation for Research and Technology - Hellas  
P.O. Box 1385, GR-71110, Heraklion, Greece  
{papagel, dp}@ics.forth.gr

<sup>2</sup> Department of Computer Science, University of Crete  
P.O. Box 2208, GR-71409, Heraklion, Greece

## 1 Introduction

Recommendation systems have 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. Content-based [1] and collaborative filtering [1], [2] algorithms are usually used to form recommendations, while additional hybrid techniques have been proposed as well. We introduce an alternative way of building user profiles based on both explicit and implicit ratings, as well as, profile-based recommendation algorithms. We advocate the use of such algorithms in order to automate the discovery of dynamic, virtual communities. As an application of the proposed techniques, we implemented MRS, a web-based information system that provides recommendations for cinema movies. The interested reader is strongly encouraged to visit the web site (<http://diat11.csd.uoc.gr>) to obtain a more detailed view.

## 2 Two Alternative User Models

Explicit ratings are considered when a specific user rates a specific item with a specific rating. We consider that the total of a user's explicit ratings comprises his profile. Explicit ratings to an item are also considered implicit ratings to the categories that this item belongs to. Implicit ratings are used to build an alternative user profile that expresses user preference to specific category of items.

## 3 “Find Buddy” and Recommendation Algorithms

### 3.1 Find Buddy Algorithms

Buddies are the users that have similar rating activity, as far as a common set of items is concerned. We employ two algorithms of calculating buddies, respectively based

on the alternative user models. Both make use of the Pearson Correlation Coefficient [3] (PCC) measure in order to calculate the relevance of two user profiles and their complexity is linear on the number of users that have rated at least once.

1. Find users that have rated at least once
2. Compare the profile of the subject user with each one of these users
  - a. Find the items that both have rated
  - b. Use PCC to calculate the relevance of the two profiles based on explicit ratings
3. If the PCC is greater than a 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

1. Find users that have rated at least once
2. Find the relevance between each user and the subject user
  - a. Take the occasional user’s profile based on genre preference
  - b. Take the subject user’s profile based on genre preference
  - c. Calculate the PCC between the two profiles
3. If the PCC is greater than a 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

### 3.2 Recommendation Algorithm Based on User’s Explicit Ratings

The following algorithm recommends items that users with similar profile like. In order to do so, it uses the “Find Buddies according to explicit ratings” algorithm to calculate buddies and puts into practice the pure collaborative filtering method [2].

1. Find the buddies of the subject user
2. Find the items that subject user has not rated yet
3. For each one of the items find the subset of buddies that have rated this item
4. Calculate the average of the ratings that this subset of buddies has given to the item
5. If the calculated average is larger than a threshold then recommend it
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 buddies) \* (max-number of not rated items)

### 3.3 Recommendation Algorithm Based on User Preference to Categories

User preference to categories can also be used to form recommendations. We propose the following algorithm in order to form recommendations exploiting the user model built by implicit ratings to categories.

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 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 preference to categories

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

The equation 1 describes the calculation performed by the algorithm for deciding whether an item 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  $i$ .

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

### 3.4 Rating Prediction Algorithm

The rating prediction algorithm guesses the rating that user  $i$  is likely to give to item  $j$ . In equation 2,  $k$  is the number of users that have rated the item  $j$ ,  $\bar{U}_j$  is the average of the user's  $i$  ratings,  $PCC_{mi}$  is the correlation between user  $m$  and user  $i$ ,  $u_{mj}$  is the rating that user  $m$  has given to item  $j$  and  $\bar{U}_m$  is the average of the  $m$ -th user's ratings.

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

The complexity of algorithm is linear on the number of users that have rated item  $j$ .

## 4 Formation of Communities

### 4.1 The "Community Coherence Grade" term

The recommendation algorithms described earlier are detecting similarities between users and they can consequently be functional in the formation of communities. In order to examine the tightness or looseness of each community we introduce the "Community Coherence Grade" (CCG) term, which is defined as the average

correlation between user  $i$  and the rest members of the community (equation 3).  $PCC_{im}$  is the calculated Pearson Correlation Coefficient of the  $m$ -th member and user  $i$  and  $k$  is the number of members of the specific community.

$$CCG_i = \frac{\sum_{m=1}^k PCC_{im}}{k}, m = 1, 2, \dots, k \quad (3)$$

In our approach communities are created automatically and are considered virtual as they exist just from the current user’s perspective and dynamic as any user’s single rating can result in an adjustment of the community’s state.

#### 4.2 Detecting Sub-communities

We introduce a filtering algorithm that facilitates the detection of users that are relevant according to just a part of their profile and consider them as sub-community. We apply criteria that typify the degree of preference to specific categories in a five-level scale (0-20, 20-40, 40-60, 60-80, 80-100 %). For example we can detect the sub-community, which likes Comedy (e.g. 80-100%) but dislikes Drama (e.g. 0-20%). The algorithm calculates the users that satisfy all the criteria by applying a Boolean AND operation to the criteria specified by the filtering procedure.

1. Find the categories for which criteria have been specified
2. For each category find which one out of the five preferences has been specified
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 calculate the positive percentage that corresponds to the user.
  - b. If the percentage satisfies the criteria then continue with the next category.
  - c. If a user satisfies the criteria of all categories then is returned by the algorithm
5. Continue with the fourth step until there are no remaining users

Algorithm 5. Applying filtering criteria to user preferences

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

#### References

1. M. Balabanovic and Y. Sholam. “Combining Content-Based and Collaborative Recommendation”. *Communications of the ACM*, 40 (3), 1997.
2. Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl, “Explaining Collaborative Filtering Recommendations”. In *proceedings of ACM 2000 Conference on Computer Supported Cooperative Work*.
3. 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.