PORS: A Peer-to-Peer Movie Recommendation System

Chian Wang and Dai-Yang Lin

***Abstract*—The rapid deployment of peer-to-peer (P2P) applications and the various shared contents have raised the problem of information overload. As a result, the users participating in a P2P network can no longer easily find the contents they really need. Recommendation systems are thus developed to suggest to the users the contents that are useful or valuable to them. But, the most of the current recommendation systems can not be easily applied to the P2P environment. In this paper, we will propose PORS (Peer-to-peer mOvie Recommendation System), which is a collaborative movie recommendation system in the P2P environment. To obtain the best recommendation results, PORS uses the download history and the ratings of watched movies of a user to make recommendation decisions. We are currently implementing the system and will have some system evaluations. We hope that PORS does provide satisfactory movie recommendations.**

***Index Terms*—Peer-to-peer, recommendation system, collaborative filtering, push, agent**

I Introduction

As the processing power of personal computers increases, the P2P architecture becomes quite popular and promising. As a result, more and more resources are available and shared in the P2P systems, such as Napster, eDonkey, and BitTorrent. Nevertheless, the huge amount of P2P users and shared contents also raises the problem of information overload. That is, it becomes harder and harder for users to search for the contents that they really need. To overcome this kind of problems, an intelligent mechanism that can help to provide a more efficient and effective searching process is preferred. Recommendation systems have been demonstrated to be one of the successful solutions [1, 2, 3]. Recommendation systems use techniques such as collaborative filtering to recommend items that are likely to be needed by a user. Collaborative filtering is a way of making intelligent predictions about a user's interests or preferences by collecting other users' evaluation or rating information. We can regard it as an "word of mouth" process of making recommendations. It tries to identify some users whose tastes are similar to a given user and then recommends the items liked by the users to the given user. However, few current recommendation systems consider the distributed nature of P2P applications.

For example, Olsson proposed a news headline recommendation system in the P2P architecture [4]. The author also solved the problem of how the peers can find each other without a centralized control. In this paper, a movie recommendation mechanism called PORS (Peer-to-peer mOvie Recommendation System) in the P2P environment is proposed. Currently, most users use movie titles as keywords to search for the desired movies in a P2P sharing system. However, if a user is not sure about the exact title or he just wants to watch a specific genre of movies, e.g., comedies starring John Travolta, it is not easy to find the movies that he wants. Worse yet, he may have to filter the search results manually to get what he really wants. Based on these considerations, a recommendation system that can help users to get the desired movies more easily and efficiently is preferred. That is, PORS is proposed to deal with the problem of information overload encountered by the users when searching for movies in a P2P sharing system. PORS is based on the concept of collaborative filtering. An important task of a collaborative filtering-based recommendation system is to build a user preference profile and to find the neighbors that have similar interests to the target user. When the neighbours are found and a neighbour set is formed to generate recommendations by referring to the neighbours' profile. The basic idea of collaborative filtering is that a user's interests and preferences do not change easily. That is, if a user has been interested in some items in the past, he is quite likely to do so now. PORS is thus proposed to help users to handle the huge amount of shared contents more efficiently and efficiently.

PORS is made up of several software agents that reside in the peers. The recommendation generation is primarily processed by the recommendation agents of each peer. PORS adopts an event-driven push mechanism that whenever the recommendation agent finds a relevant movie shared in the system, it pushes a notification message to the user. PORS also employs a filtering based on the recent download history of each peer. We believe that the download history is a good observation on a user's interest. Our goal is that PORS can provide a pleasant search experience in the P2P system.

II Personal Recommendation Agents

As mentioned previously, PORS is developed to help users to search for movies more efficiently and effectively in a P2P system. PORS consists of several interconnected software agents that reside on each peer. The primary agents include the event-driven push agent, the download history filtering agent, and the web page retrieval agent. The event-driven push agent forwards information about newly shared movies to the user. The download history filtering agent selects movies for recommendation among pushed movies from the event-driven push agent. The web page retrieval agent retrieves movie information from the web sites. For PORS to work properly, it is required that each peer keeps the user's download history and he is willing to share downloaded movies. The recommendation agents collaborate by exchanging the user preference information.

One of the challenging problems of recommendation systems is how to obtain user preference information

Manuscript received April 13, 2012; revised May 24, 2012.

The authors are with Department of Information Management, National Changhua University of Education, Changhua, Taiwan (Tel: +886-4-7232105 ext 7330; fax: +886-4-7211295. e-mail: cwang@cc.ncue.edu.tw)

because it is an important basis to generate satisfactory recommendations. A simple approach to collect user preference information is to have each user explicitly specify his preferences about a specific item. However, this approach requires additional user efforts when accessing the items, which is not practical in a P2P system. Another approach is to learn a user's preference by referring to his download history. We employed the later approach in PORS. That is, implicit preferences are not inputted directly by a user, but are inferred from observation of the user's download history. When developing PORS, some data mining techniques are adopted for this purpose. We are also implementing a model-based algorithm which generates a descriptive model by compiling the user's preferences and then the movie recommendations are generated by appealing to the model.

In PORS, each user's preference information is represented by a recommendation model. The recommendation model consists of the host peer profile and the target peer set. The download history filtering agent keeps watching over its host peer and learns what types of movies the user downloads more, which is then presented in the host peer profile. That is, whenever the user downloads a movie, the download history filtering agent of the peer keeps information about the movie with the aid of the web page retrieval agent. This knowledge is then used as the basis to generate movie recommendations. The event-driven push agent pushes the recommendations to the target peer set.

III The Recommendation Model

As mentioned above, the agents build an individual recommendation model and exploit it locally. The construction of the recommendation model is an essential task since PORS works heavily based on the recommendation model. In order to build the recommendation model, the agents consider the movie genre that the user prefers represented in the host peer profile. In addition, using PORS, a user rates each movie he watched to represent his evaluation and interest about the movie. When two users approach, PORS exchanges the rating information stored on each user's peer to determine if they have similar interests. When a user's interest is determined or matched, PORS generates a recommendation list containing movie information, e.g., title, genre, director, stars, plot, etc. There is a data table on each user's peer. The table is used to store the user's preference and the ratings of the movies that he watched. There are two parts of the rating data, as illustrated in Table I.

TABLE I: The Rating Data Table.

f1	f2	f3	f5	tg1	tg2	tg3	tg4	tg5
8	7	2	4	4	2	8	3	9

In Table I, f_i means the user's rating of movie i and tg_i represents his preference of movie genre i. The ratings are from 1 (the lowest) to 10 (the highest). When two users approach, the ratings stored in their tables are exchanged in order to generate movie recommendations. There are three ways of exchange.

(1) Unlimited exchange. Whenever two users get into a specific range, all the items in their data tables are exchanged. This is the simplest way of exchange. However, some items may be duplicate if they watched the same movies before. Moreover, as a user meets more other users, the amount of data increases rapidly.

(2) Limited exchange. In order to prevent the problem of flooding, each rating can be associated with a time-to-exchange (TTE) value. When a rating is exchanged, its TTE is decreased by one. When the TTE becomes 0, the rating can not be exchanged anymore.

(3) Exchange with similarity. In this paper, we adopt Pearson correlation to measure the similarity of user interests. If two users' interests are similar, then their rating data can be exchanged. By considering user interests, the recommendation results can be more accurate and acceptable.

Using the proposed mechanism, the similarity of two users' interests $W_{A,B}$ is determined as follows.

$$W_{AB} = \frac{\sum (r_{Ai} - \overline{r_A})(r_{Bi} - \overline{r_B})}{\sqrt{\sum (r_{Ai} - \overline{r_A})^2 (r_{Bi} - \overline{r_B})^2}}$$

where rAi is user A's rating of movie i,
rBi is user B's rating of movie i,
rA is the average of A's ratings, and
rB is the average of B's ratings.

WA,B is between 1 and -1. If WA,B is closer to 1 (-1), it means that A and B's interests are quite similar (different). If WA,B is larger than a specific threshold, TH, then A and B's rating data can be exchanged. Otherwise, their rating data are not exchanged in order to prevent the problem of flooding.

For example, Tables II (a) and II (b) show the rating data of A and B, respectively.

TABLE II: The Rating Data of Users A and B.

f1	f2	f3	f6	tg1	tg2	tg3	tg4	tg5
8	7	1	4	4	2	9	1	9

(a)

f1	f2	f3	f5	f6	tg1	tg2	tg3	tg4	tg5
7	9	3	9	2	3	2	7	3	8

(b)

Then, their $W_{A,B}$ is 0.8322. If *TH* is set to 0.6, then A and B can exchange their rating data.

It is quite obvious that the effectiveness and success of a recommendation system depends on the ability to represent the user's actual preferences. The download history and movie ratings of a user are a useful basis to deduce his movie preference. Also, each movie is represented as a collection of attributes that are used to describe its properties used for generating recommendations. As a user's download history and preferences change over time, PORS utilizes the peer profiles to reflect the current preferences in real time. That is, the peer profiles are updated whenever relevant movies are newly downloaded to reflect the user's most recent preferences.

IV The Overall Operations of PORS

PORS recommends only the selected movies to the user by filtering the movies shared in the P2P system. It achieves this goal by having two levels of the filtering process. The first level is the peer-based filtering that filters the user's download history and the rating data table to deduce his preferences. The second level is the agent-based filtering that selects newly shared movies in the P2P system for recommendation based on the match results with the user's preferences. These two levels of the filtering process are done iteratively by the three agents mentioned previously: the event-driven push agent, the download history filtering agent, and the web page retrieval agent.

When PORS makes recommendation decisions, another agent, the top-k filtering agent is triggered. Whenever a user finishes downloading a movie, his download history is updated. If a newly shared movie is matched with the user's preferences, its information is added to the user's recommendation queue. Then, the top-k filtering agent processes the movie set in the recommendation queue by selecting the top-k movies with higher enjoy likeliness score obtained implicitly during the matching process. That is, the top-k filtering agent selects k movies from the recommendation queue to generate a recommendation list. The filtered movies are accumulated sequentially until the queue is full. When this happens, the top-k filtering agent removes the movies orderly from the queue's beginning to accommodate more newly filtered movies. The top-k filtering agent makes recommendation decisions based on the enjoy likeliness score, denoted as ELS(h, i) of user h on movie i. The top-k filtering agent generates a list of k movies, $R=\{r_1, r_2, r_3, ..., r_k\}$ such that r_i is not in the set of movies that user h has already download and ELS(h, r_1) is the highest, ELS(h, r_2) is the second highest, and so on. After this, the recommendation movie list is generated and is pushed to the user by the event-driven push agent. Then, the user browses through the recommendation list to see if there are any movies of his interest. The user may select one movie for download and makes a rating decision about the movie. The download history and the rating data table in the recommendation model are then updated accordingly.

V Conclusion

Currently, we are developing the corresponding application system and having a real-world system evaluation. Our evaluation will be based on several criteria, including Mean Absolute Error (MAE) and precision. MAE's purpose is to evaluate the difference between the real and the estimated ratings. If MAE is smaller, it means that PORS can generate more satisfactory recommendations. On the other hand, precision's purpose is to measure the correctness of the estimations. Precision is between 0 and 1. When precision is closer to 1, the estimation is more satisfactory. Otherwise, the estimation is less correct. Hopefully, PROS can also be applied to other areas.

References

[1] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon and J. Riedl, "GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM*, vol. 40, no. 3, pp.77–87, March 1997.

[2] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter, PHOAKS: A System for Sharing Recommendations. *Communications of the ACM*, vol. 40, no. 3, pp. 59–62, March 1997.

[3] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating Word of Mouth," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 210-217, 1995.

[4] Olsson, T. Bootstrapping and Decentralizing Recommender Systems. Ph.D. Thesis, Department of Information Technology, Uppsala University, 2003.