Technical Report CS-2007-37

Recommender Schemes for Peer-to-Peer Systems

Loubna Mekouar
University of Waterloo
Waterloo, Canada
Email: lmekouar@bbcr.uwaterloo.ca

Youssef Iraqi
Dhofar University
Salalah, Oman
Email: y_iraqi@du.edu.om

Raouf Boutaba
University of Waterloo
Waterloo, Canada
Email: rboutaba@bbcr.uwaterloo.ca

***Abstract*— In Peer-to-Peer (P2P) file sharing systems, peers spend a significant amount of time looking for relevant and interesting files. However, the files available for download represent on one hand a rich collection for different needs and preferences and on the other hand a struggle for the peers to find files that they like. In this paper, we propose new recommender schemes based on *collaborative filtering*. Peers collaborate to filter out irrelevant files. These schemes help peers find and discover new, interesting and relevant files. We propose recommender schemes based on *Files' Popularity* and/or *Peers' Similarity*. To overcome the problems of traditional collaborative filtering recommender systems, an implicit rating approach is used. Simulation results confirm the effectiveness of *Peers' Similarity based Recommendation* in providing accurate recommendations. In addition, the proposed recommender schemes are proactive. Recommendations are provided to peers to motivate them to download the recommended files.**

I. Introduction

Since Peer-to-Peer (P2P) file sharing systems have emerged as a new way to share files, users[1] are overwhelmed by a huge selection of files available for download. These files may be songs, movies, software, or books. Unfortunately, users have to struggle to choose the right items that are of interest to them. Now and with so many P2P systems to choose from, users are more demanding. It is not enough for users to get files, but they require to receive personalized recommendations as added value. Using a recommender system will help users to search through this very large selection of files. Based on peers' behavior, the recommender system will suggest to them items that they will most probably like. These peers will be motivated to download the recommended files and hence, will remain active members. While these peers are downloading the files, they will serve other peers by uploading files to them. Moreover, the recommender system will also attract more users to P2P file sharing applications since from the user's perspective, the search process for interesting files will be easier and more efficient. New recommender mechanisms are needed to filter the available information and find files that are most valuable to peers.

Recommender systems are widely used in E-commerce applications (e.g. amazon.com and CD-now.com) to provide customers with buying recommendations [1], [2], [3]. The recommender systems suggest products and services that most likely will be of interest to the customers. These systems take advantage of the collected data that represents customers' experiences, choices and profiles to predict future needs of these users.

Although, E-commerce applications have been using and benefiting from recommender systems for a long time, recommender schemes are still a fertile area in peer-to-peer environments. Only few research works have been proposed in this area. The proposed recommender schemes in [4], [5] are suitable for decentralized P2P systems (e.g. Gnutella [6]) but not for the partially decentralized systems (e.g. KAZaA [7], Morpheus [8], eDonkey and Gnutella2 [9]). However, partially decentralized systems are the most popular and to our knowledge no recommender scheme has been proposed for these systems.

In this paper, we propose new recommender schemes for partially decentralized P2P systems.

The search process in partially decentralized P2P file sharing systems can be divided into the following steps:

[1]Through out the paper, "users" and "peers" are used interchangeably

1) Providing interesting keywords.
2) Sending the request to the supernode.
3) The supernode will search for the file by contacting local peers if the file is available locally or sending a request to other supernodes.

The goal from using a recommender system is to save the peer's effort in the first step by finding relevant filenames for files of interest based on the peer's profile that reflects her/his past choices, experience and preferences.

The paper is organized as follows. Section II describes the recommender schemes used in E-commerce. Section III describes the proposed recommender schemes. Section IV presents the performance evaluation. Finally, Section V concludes the paper.

II. Recommender Systems in E-Commerce

A. Content-based versus Collaborative Filtering

Most recommender systems in e-commerce are based on one of the two following techniques [1], [10], [3], [4]: content-based and collaborative filtering.

In the content-based approach, items are described in terms of their characteristics and recommendations are based on items already selected by the active user (i.e. the user to whom a recommendation is made). This approach is time consuming and expensive for subjective files such as songs and movies. It is not recommended for highly dynamic environments with millions of users and files [10].

Collaborative filtering is the most widely used technique for recommender systems. This approach is based on collecting users' ratings. It suggests items based on similarities between the active user's profile and other users or similarities between items. In this approach, it is required that a large number of users rate items to ensure recommendation accuracy [10]. This technique has proved to be one of the most successful techniques in recommender systems in recent years.

B. Challenges of Collaborative Filtering Algorithms

The main possible problems of collaborative filtering are the followings [1], [10], [3], [4]:

- *Cold start*: This problem occurs for a new user or at the start of the system. It is difficult to make recommendations for a new user based on users' similarities since no rating is provided yet and the user's profile is not known yet.
- *Popularity effect*: This problem occurs when the given recommendations are obvious and evident from the user's point of view.
- *Data sparseness*: This problem occurs when only few users have rated few items. It is difficult to predict the user's interests and make accurate recommendations.
- *Trust*: This problem occurs when untrustworthy users provide false ratings. The system should be able to choose only highly reputable users while making recommendations. This will reduce the impact of untrustworthy users that influence badly the recommendation accuracy and hence, will increase the trust given by the peers to the recommender system.

III. The proposed Recommender Schemes for Partially Decentralized P2P Systems

A. Explicit Rating versus Implicit Rating

In E-commerce, the collaborative filtering technique is based on the ratings by the customers of the products and/or the purchases they made. In P2P systems, the collaborative filtering technique can be used based on the ratings by the users of the files they have. We can distinguish two approaches to rating: *explicit rating* and *implicit rating*.

In the *explicit rating* approach, the user has to explicitly provide a rating for each file he/she downloads according to its content (i.e. matches the user's preferences or not). This approach necessitates additional effort from the users. A rating scheme from 1 (not interesting at all) to 5 (very interesting) can be useful to assure recommendation accuracy. Users have to provide their ratings for different files (e.g. movies and songs that they have downloaded) to enrich the system with different opinions and experiences. Since *explicit rating* solicits an additional effort from users, that can not be guaranteed in systems where 70% are free riders[2], it is difficult to enforce explicit rating. This approach will likely suffer from the *cold start* and *data sparseness* problems. Also, *explicit rating* provides malicious peers with a way to influence the rating system which may lead to the *trust* issue as discussed in section II-B. The *implicit rating* approach does not require the users to rate the files. It assigns ratings implicitly (i.e. automatically without involving the user) based on users' profile. We propose to assign a rating of 1 (*I like it*) to the files owned by the user. All other files are assigned a rating of 0 (*I do not know*). Note that a rating of 0 does not mean that the user does not like the file.

[2]Free riders are peers that take advantage of the system without contributing to it. They do not upload files to other peers even if they are downloading from them

The fact that ratings are generated automatically without involving users, alleviate them from the burden of providing ratings for each file they have downloaded.

The *implicit rating* approach has the following advantages:

- It solves the cold start problem: Indeed, even at the start of the system or when a new user joins the P2P system, ratings are available.
- It avoids the *data sparseness* problem as a rating of 0 or 1 is automatically available for each and every file in the system.
- The subjective rating of files open the door to malicious peers to manipulate files' recommendation. The implicit rating approach avoids tempering with the ratings of the files by malicious peers which reduces their impact on the recommender system and thereby avoiding the *trust* problem.
- It avoids the *popularity effect* problem: No recommendation is given for files already owned by the user.

For all these reasons, we adopt the *implicit rating* approach.

B. User-based Collaborative Filtering versus Item-based Collaborative Filtering

In this paper, we consider partially decentralized P2P systems. In these systems, peers connect to their supernodes that index shared files and proxy search requests on behalf of these peers. Queries are therefore sent to supernodes, not to other peers. Once the peer sends a search request to its supernode, this latter will forward this query to other supernodes. The supernode will receive information regarding peers that have the requested file, and a list of files that they are willing to share.

Figure 1 depicts an example of the information flow between the peer P_1 requesting a file and its supernode. After receiving a request from peer P_1, and assuming the file is not found locally, its supernode sends a request to other supernodes. These supernodes will send back the search result which is a list of peers that have the requested file in addition to the files that these peers are sharing. Based on this information, the supernode of P_1 will use the proposed recommender schemes to generate a list of recommended files.

Partial search performed by supernodes limits the number of peers that have the requested file in the search result. This number is much less than the number of files shared by all the peers in the system. In addition, finding relationships between all files is time consuming

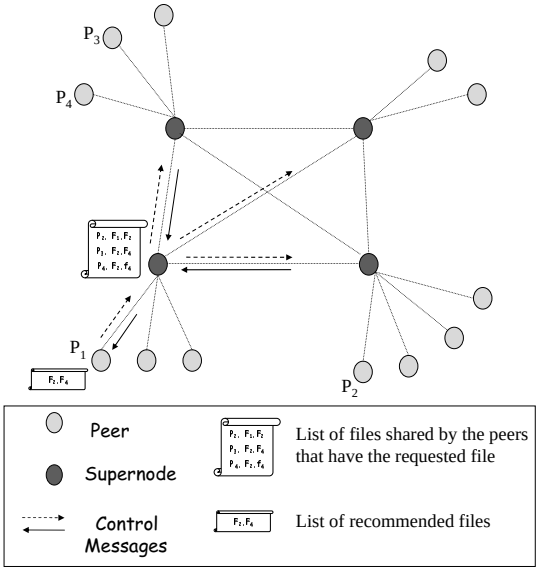

Fig. 1. Example of the information flow

and is usually done offline. But, since recommendations are given to the peer in real time, it is preferable to explore relationships between peers rather than between files. For these reasons, using user-based collaborative filtering in P2P systems is more practical than using item-based collaborative filtering algorithms.

C. Formal Notations

In the remaining of the paper, we will use the following formal notations:

Let P be the set of all peers in the system.

Let F be the set of all files shared by the peers in the system.

Let p_i be the requestor peer looking for a file f_x. p_i is the peer of the user to whom the recommendation will be made.

Let P_{f_x} be the set of peers that possess the file f_x.

Let $F_{P_{f_x}}$ be the set of files that these peers possess in addition to f_x. This is the set of files that peers are willing to share.

Let $f : P \rightarrow \Omega(F)$, such that $f(p_j)$ is the set of files held by peer p_j for every j and $\Omega(F)$ is the power set of F. Then we have:

$$F_{P_{f_x}} = \bigcup_{p_k \in P_{f_x}} f(p_k)$$

D. Files' Popularity Based Recommendation (FP)

To avoid the *popularity effect* problem, we do not consider the files that p_i possesses. This technique will

allow a peer to discover the files that are more popular within the peers that have the requested file.

Let $G_{p_i} = F_{P_{f_x}} - f(p_i) - \{f_x\}$ be the set of files that p_i does not have from the set $F_{P_{f_x}}$ not including the file f_x. These are all the files owned by the peers in P_{f_x} that the peer p_i does not have.

For every file $f_k \in G_{p_i}$, we define the popularity of the file as:

$$Pop(f_k) = \frac{|P_{f_k} \cap P_{f_x}|}{|P_{f_x}|}$$

where $|P|$ is the cardinal of set P.

The value of $Pop(f_k)$ is a numerical score that shows the likeliness of the file f_k among the peers in P_{f_x}.

In this technique, files that are more popular will be recommended. This scheme will recommend only files f_k such that $Pop(f_k) \geq t_1$, where t_1 is a threshold. This recommendation list is sorted according to the popularity of the files $Pop(f_k)$ with the files that are most popular at the top of the list. The supernode of peer p_i may keep track of these files for future recommendations. This technique will accelerate significantly the spread of popular files which will increase peers' satisfaction.

E. Asymmetric Peers' Similarity Based Recommendation with File Popularity (ASFP)

Peers' similarity is an important factor in this technique. To be able to make accurate recommendations, we compare the active user's files against those of other users. The goal of this process is to find peers with similar preferences as the active peer p_i and make recommendations on the files that they have. In fact, we apply the files' popularity approach within these peers.

For every peer p_j in P_{f_x} we define the similarity relationship as:

$$ASim_{p_i}(p_j) = \frac{|f(p_i) \cap f(p_j)|}{|f(p_i)|}$$

We assume that $|f(p_i)|$ is not null, which means that the peer p_i owns at least one file. If the peer does not own any file, the *FPR* scheme is used.

The value of $ASim_{p_i}(p_j)$ is a numerical score that shows how similar the peer p_j is to the peer p_i. Note that this similarity relationship is not symmetric, i.e. $ASim_{p_i}(p_j)$ may not be equal to $ASim_{p_j}(p_i)$

This scheme will choose only peers that have $ASim_{p_i}(p_j) \geq t_2$. Where t_2 is a threshold.

Let $S_{p_i}^{t_2} = \{p_j, p_j \in P_{f_x} \; and \; ASim_{p_i}(p_j) \geq t_2\}$

We apply the *FPR* within the set $S_{p_i}^{t_2}$ of peers most similar to peer p_i.

For every file, we compute

$$Pop_{ASim}(f_k) = \frac{|P_{f_k} \cap P_{f_x} \cap S_{p_i}^{t_2}|}{|P_{f_x} \cap S_{p_i}^{t_2}|}$$

Note that if $t_2 = 0$ then $Pop_{ASim}(f_k) = Pop(f_k)$.

This scheme will recommend only files f_k such that $Pop_{ASim}(f_k) \geq t_1$, where t_1 is a threshold. This recommendation list is sorted according to the popularity of the files $Pop_{ASim}(f_k)$ with the files that are most popular at the top of the list. Both t_1 and t_2 are application dependant values. The higher these values are the more exigent we are in recommending files to the peers in the system.

The proposed schemes are proactive. After sending a search request for a file, the peer receives recommendations automatically. The peer will be tempted to download the recommended files and hence, increasing the peer's contribution [11]. This is one of the advantages of the proposed schemes.

IV. Performance Evaluation

A. Simulated Schemes

We have simulated the following techniques:

- *Random Based Recommendation (RBR)* where peers will choose files randomly. Although users usually tend to search and download files according to their choices and needs, we wanted to see the effect of the recommender on the P2P system.
- *Files' Popularity Based Recommendation (FP)* (see section III-D).
- *Asymmetric Peers' Similarity Based Recommendation with File Popularity (ASFP)* (see section III-E).

B. Simulation Parameters

The simulation parameters are the following:

- We simulate a system with 1,000 peers and 1,000 files.
- At the beginning of the simulation, each peer has at most 50 files and each file has at least one owner.
- In our simulations, the first file in the list of recommended files is always chosen. However if no file is recommended, file requests follow the real life distribution observed in [12]. Each peer can ask for a file with a Zipf distribution over all the files that the peer does not already have. The Zipf distribution parameter is set to 0.9 (as in [13]).
- Peers are divided into four interest categories (C1: Action, C2: Romance, C3: Drama and, C4: Comedy) and files are also divided into the same four categories. Each peer belongs to one category. For

this reason, peers prefer to have most of the files from their category and only few files from other categories. At the beginning of the simulations, each peer will get files from the category that she/he prefers with a probability of 0.9 and files from other categories with a probability of 0.1.

- The percentage of peers in each category is 25%.
- The percentage of files in each category is 25%.
- The maximum number of files proposed in each recommendation is set to 5. We also keep history of previously recommended files.
- To assess the performance of the considered schemes in a highly dynamic environment, only 60% of all peers with the requested file are found in each search request since only partial search results are obtained in partially decentralized P2P systems.
- The threshold t_1 is set to 0.1. This means that the popularity of a file should be greater than 10% for the file to be considered for recommendation.
- The threshold t_2 is set to 0.2. This means that the similarity of a peer should be greater than 20% for the peer to be considered.
- We simulate 200,000 requests for each simulation. Simulations are repeated 10 times and the results presented are the average values.

Following the simulation parameters, peers with indices from 1 to 250 belong to the category C1 (Action), peers with indices from 251 to 500 belong to the category C2 (Romance). Peers with indices from 501 to 725 belong to C3 (Drama) and peers with indices from 726 to 1,000 are peers that belong to C4 (Comedy).

C. Performance Metrics

For each scheme, we computed the following performance metrics:

- For each peer's category, we compute the ratio of the number of downloaded files that belong to this category over all the files downloaded by the peers from this category.
- For each peer's category, we compute the ratio of the number of downloaded files that belong to other categories over all the files downloaded by the peers from this category.

D. Simulation Results

Figure 2 depicts the percentage of files for each peer category based on the random selection scheme. The X axis represents the four peers' categories while the Y axis represents the percentage of files for each peer category. In the X axis, the (a) bars show the

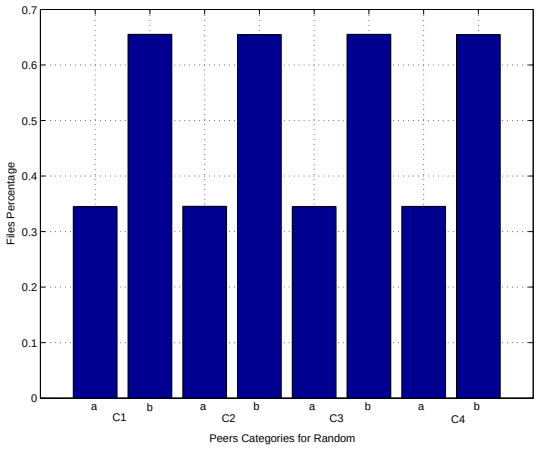

Fig. 2. Percentage of files when using Random selection of files

percentage of files from the same category as the peers' category while the (b) bars show the percentage of files from other categories. Peers from different categories choose files randomly. No recommendation is provided. As expected, when choosing files randomly, peers from different categories have similar percentages of files from the category that they prefer (about 35% as shown by the (a) bars) and from other categories (about 65% as shown by the (b) bars). These numbers are expected since at most 50 files are distributed to each peer at the beginning of the simulation with 90% of them from the peer's category and 10% from other categories. And during the simulations about 200 files are downloaded by each peer from random categories. This means that 25% of the 200 files will belong to the peer's category and 75% of them will belong to other categories. So in total, the peer should have about 95 ($50 \times 90\% + 200 \times 25\%$) files from her/his category and about 155 ($50 \times 10\% + 200 \times 75\%$) from other categories.

Figure 3 depicts the percentage of files for each peer category when using the FP scheme. In the first peer category (C1), peers possess 21% of files from the category that they prefer (a) and 79% from other categories (b). This technique shows that Files Popularity does not fulfill peers expectations and does not match their preferences. However, the benefit from this scheme is that peers are aware of the most popular files within the file sharing system and this scheme is effective in spreading popular files.

Figure 4 depicts the percentage of files for each peer category based on the ASFP scheme. This figure shows clearly the effectiveness of this recommender scheme. In the first peer category (C1), peers have 89% from the

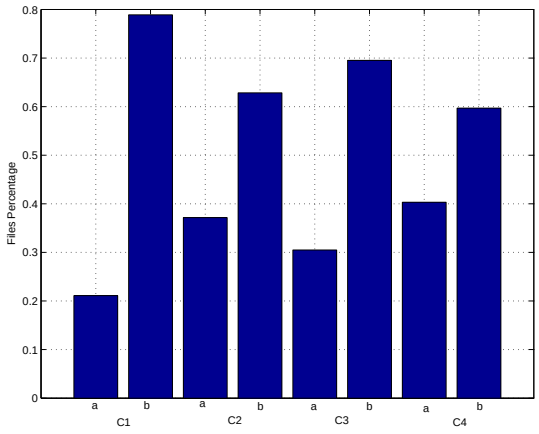

Fig. 3. Percentage of files when using FP

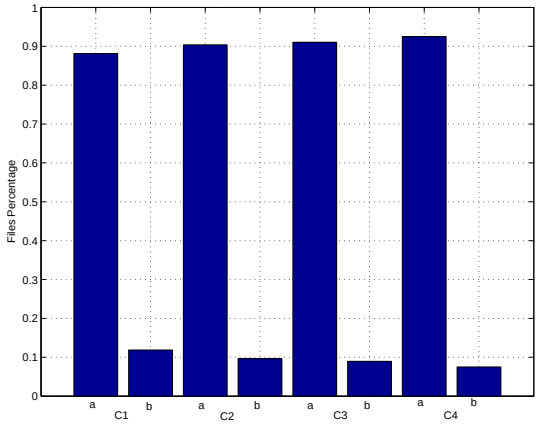

Fig. 4. Percentage of files when using ASFP Scheme

first category of files that they like (a) and only 11% from all other files categories (b). From this figure, it is clear that peers have more files from the category that they prefer. This is shown by the fact that (a) bars are higher than (b) bars for all categories.

In these simulations, the thresholds values for the ASFP scheme are not null. t_1 and t_2 are set to 0.1 and 0.2 respectively. This is important in providing accurate recommendations since a null value for t_2 will lead to considering all the peers that have the requested file as similar peers to the active peer. All these peers will be considered as neighbors to the active peer which is not necessarily true. On the other hand, a high value for these thresholds will make the recommender scheme very exigent in providing recommendations. In this case, it is possible that no recommendation is made. Files that will be requested by a peer will follow a Zipf distribution over all the files that this peer does not already have. The requested files do not necessary belong to the peers' category.

V. Conclusion

In this paper, we proposed new recommender schemes for partially decentralized peer-to-peer systems. The proposed recommender schemes help the peers discover relevant files, this way, increasing peers satisfaction. While the peers are downloading these files, they will remain connected and upload files to other peers. The proposed recommender schemes are proactive, easy to understand by users and easy to implement and maintain. Moreover, the proposed schemes use implicit rating and do not suffer from the problems that traditional collaborative filtering schemes suffer from like the cold start and the data sparseness problems. Simulation results show the effectiveness of the *Asymmetric Peers' Similarity Based Recommendation with File Popularity* scheme in providing accurate recommendations and hence, increasing the trust given by the peers to the recommender scheme.

References

[1] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Analysis of Recommendation algorithms for E-commerce," in *EC*, October 2000.

[2] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," in *IEEE Internet Computing*, Jan/Feb 2003, pp. 76–80.

[3] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-Based Collaborative Filtering Recommendation Algorithms," in *WWW*, May 2001.

[4] J. Wang, J. Pouwelse, R. L. Lagendijk, and M. J. T. Reinders, "Distributive Collaborative Filtering for Peer-to-Peer File Sharing Systems," in *SAC*, April 2006.

[5] G. Ruffo, R. Schifanella, and E. Ghiringhello, "A Decentralized Recommendation System based on Self-Organizing Partnerships," in *IFIP-Networking 2006*, May 2006, pp. 618–629.

[6] Gnutella, "www9.limewire.com/developer/gnutella_protocol_0.4.pdf," .

[7] KaZaA, "http://www.kazaa.com/," .

[8] Morpheus, "http://morpheus.com," .

[9] Gnutella2, "http://www.gnutella2.com/," .

[10] P. Massa and Paolo avesani, "Trust-aware Collaborative Filtering for Recommender Systems," in *International Conference on Cooperative Information Systems (CoopIS)*, - 2004.

[11] L. Mekouar, Y. Iraqi, and R. Boutaba, "Free Riders under Control through Service Differentiation in Peer-to-Peer Systems," in *IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2005)*, December 2005.

[12] K. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and analysis of a Peer-to-Peer File Sharing Workload," in *19th ACM Symposium on Operating Systems Principles*, 2003.

[13] L. Mekouar, Y. Iraqi, and R. Boutaba, "Peer-to-Peer Most Wanted: Malicious Peers," *The Computer Networks Journal, Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security*, vol. 50, no. 4, pp. 545–562, 2006.