Evaluation of P2P Information Recommendation Based on Collaborative Filtering

Hidehiko Okada and Makoto Inoue

Kyoto Sangyo University
Kamigamo Motoyama, Kita-ku, Kyoto 603-8555, Japan
hidehiko@ics.kyoto-su.ac.jp

Abstract. Collaborative filtering is a social information recommendation/ filtering method, and the peer-to-peer (P2P) computer network is a network on which information is distributed on the peer-to-peer basis (each peer node works as a server, a client, and even a router). This research aims to develop a model of P2P information recommendation system based on collaborative filtering and evaluate the ability of the system by computer simulations based on the model. We previously proposed a simple model, and the model in this paper is a modified one that is more focused on recommendation agents and user-agent interactions. We have developed a computer simulator program and tested simulations with several parameter settings. From the results of the simulations, recommendation recall and precision are evaluated. Findings are that the agents are likely to overly recommend so that the recall score becomes high but the precision score becomes low.

Keywords: Multi agents, P2P network, information recommendation, collaborative filtering, simulation.

1 Introduction

Several information recommendation and filtering methods have been developed for solving the problem of information overloads [1-3]. One of the methods is collaborative filtering [4]. The method can discover information that a user does not yet know but other users (whose profiles are similar to the user in some point of view) know. An advantage of the method over content-based filtering methods is that collaborative filtering does not require analyses of information contents: precise semantic analysis of information contents is usually difficult for computer systems. Collaborative filtering can be applied to personalized recommendations. There will be two variations of such recommendation systems that utilize collaborative filtering on computer networks: a server-client based one and a peer-to-peer (P2P) based one [5-7]. The latter benefits its scalability and flexibility.

To evaluate effectiveness of an application system on a large-scale P2P network, a large number of nodes must participate in the evaluation, but obtaining the participation and cooperation of the large number of nodes in a short period of time is usually difficult. Instead of evaluating the system in the real world, it will be effective to pre-evaluate the system by developing a simulator. Related researches have developed P2P network simulators (e.g., [8]), but a simulator for P2P information distribution systems based on collaborative filtering is still a research challenge. We previously proposed a model of P2P information recommendation system based on collaborative filtering [9]. In this paper, we propose a modified model that is more focused on recommendation agents and user-agent interactions. We have developed a computer simulator program to evaluate the ability of recommendation agents based on our model. By using the program, we have tested simulations with several parameter settings. From the results of the simulations, the ability of agents is evaluated.

2 P2P Information Recommendation Model Based on Collaborative Filtering

The basic idea of our simulation model is as follows. First, a P2P network includes several nodes (computers), and a user uses a computer that works as a node in the P2P network (Fig.1). Each user periodically receives recommendations of some data items from an agent that serves for the user. An agent determines which items to recommend by collaborative filtering with some neighbor nodes. Of the items recommended by an agent, a user accepts those which meet his/her preference and rejects the others.

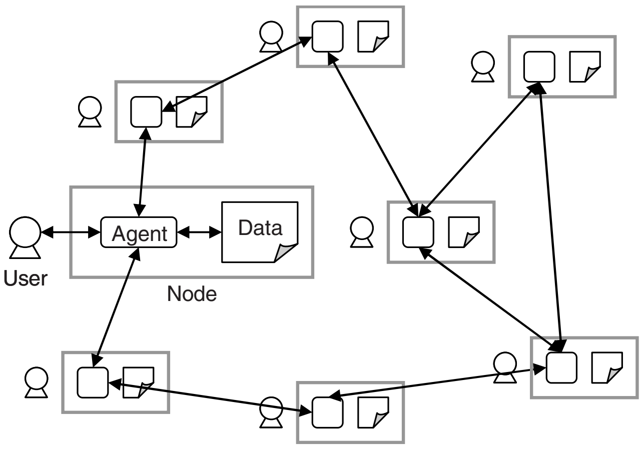


Fig. 1. Nodes, Users and Agents in a P2P Network

Based on this idea, our system model consists of the following components.

- Network model
- Data model
- Agent model
- User model

2.1 Network Model

Suppose a P2P network consists of N nodes (computers). Each node has a list of neighbor nodes with which the node can communicate. The node lists are updated when agents communicate with each other: when an agent A1 of a node N1 communicate with another agent A2 of another node N2, A1 (A2) merges the node list of A2 (A1) node with the node list of its own node. Fig. 2 shows an example. The left part of the figure shows that agent A1 initiates to communicate with agent A2 (A1 can find A2 because N2 is included in the node list of N1). The right part of the figure shows that node lists of N1 and N2 are updated: the underlined nodes are added from the list of partner node. A2 adds N1 in the list of N2 because A2 detects N1 by this communication and N1 is not included in the list of N2. As shown in this example, an agent can find new nodes on the network each time the agent communicates with another agent. Thus, the agent becomes able to search for data to recommend from more neighbor nodes.

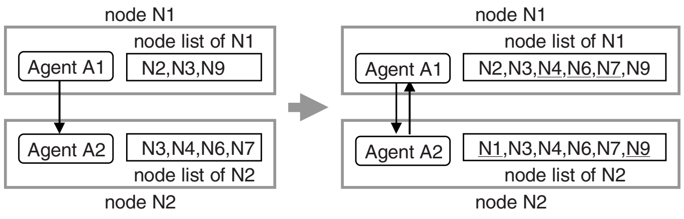


Fig. 2. Example of Node Lists Updated by Agents

2.2 Data Model

Suppose there are D data items in total, and each item belongs to a data category with some degree. In the case where data items are musical ones, the categories can be blues, classical, pop, jazz, etc. Table 1 shows an example of data items and membership scores of the items to data categories. In this example, date item D1 belongs to data category C1 completely and not to C2 and C3 at all: the membership scores are in [0, 1] and the higher the score is, the more the data item belongs to the data category.

The vector of category membership scores is attached to each data item as metadata.

2.3 Agent Model

The most important component in our system model is the agent model, because the design of agents determines the ability of the recommendation system. Each node includes a recommendation agent that serves for the user of the node. Each agent periodically recommends data items determined by collaborative filtering with some neighbor nodes.

Table 1. Example of Data Items and Their Membership Scores to Data Categories

Data Item	Data Category			
	C1	C2	C3	…
D1	1.0	0.0	0.0	…
D2	0.0	1.0	0.5	…
D3	1.0	1.0	0.0	…
…	…	…	…	…

An agent of a node first selects some nodes in the current node list. The nodes can be selected, for example, randomly, based on the past selection history, or based on similarity scores (described next) at the last communication. Then, the agent communicates with each agent in the selected nodes, checks the data in each selected node, and calculates similarity score between its own node and each selected node. The similarity scores are used for the collaborative filtering: the more similar set of data items a neighbor node has, the more probable the data items includes useful ones for the user of its own nodes (i.e., the more probable the data items includes items that should be recommended for the user). We define the similarity score $S(N_a, N_b)$ between two nodes N_a and N_b as

$$S(N_a, N_b) = \frac{2|D_a \cap D_b|}{|D_a| + |D_b|} \quad . \tag{1}$$

where D_a and D_b denote the set of data items in the node N_a and N_b respectively and $|X|$ denotes the number of data items in the set X. The score S becomes the largest 1.0 in the case the two nodes have the same data items and becomes the smallest 0.0 in the case $D_a \cap D_b = \Phi$.

The agent then extracts data items that the selected nodes have but its own node does not yet have, and determines which of the extracted items to recommend based on a probability defined as a function of the similarity score: a data item found in a neighbor node with a larger/smaller similarity score is recommended with higher/lower probability. Fig. 3 shows examples of the recommendation probability function. In the case of (a), an item found in a node with similarity score x is recommended with the probability x. In the case of (b) and (c), an item found in a node with smaller similarity is recommended with much less probability (zero if $S < 0.5$ in the (c) case). Thus, the design of this function characterizes agent recommendation behavior.

2.4 User Model

Each user periodically receives recommendations of data items (that the user does not yet have) from an agent that serves for the user. In the real world, a user will accept some of the recommended data items that meet his/her preference (the accepted items are added to the data set of his/her own nodes) and will reject the others. To simulate this user behavior, users' implicit preferences on the data items are denoted as preference score vectors in our user model. The user preference vectors are similar to

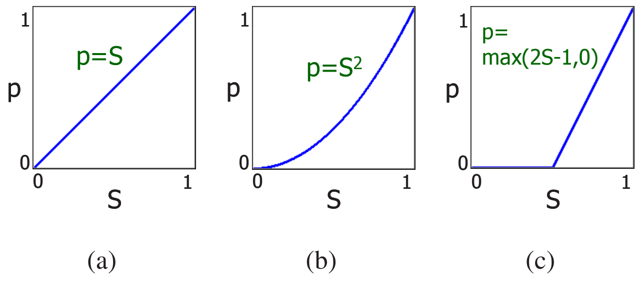


Fig. 3. Examples of Recommendation Probability Function

Table 2. Example of Users and Their Preference Scores on Data Categories

User	Data Category			
	C1	C2	C3	…
U1	1.0	0.0	0.0	…
U2	0.0	0.3	0.8	…
U3	0.2	1.0	0.0	…
…	…	…	…	…

the category membership vectors of data items. Table 2 shows an example of users and preference scores of the users on data categories. In this example, user U1 prefers data category C1 to the maximum degree and not C2 and C3 at all: the preference score is in [0, 1] and the higher the score is, the more the user prefers the data category.

Note that the user preference vector of each user is *implicit* so that an agent does not know the preference vector of the user: our model of recommendation system does not require users to express (i.e., input to the system) their preferences on data categories. The user preference vectors are used only to simulate the user behavior of data item acceptance/rejection.

Based on the preference vector of a user and the membership vector of a data item, a distance value of the two vectors is calculated to evaluate the degree to which the data item meets the user's preference. The method of calculating the distance characterizes users' personality. Suppose the user U1 in Table 2 receives the recommendation of data items D1, D2 and D3 in Table 1.

- Suppose U1 minds whether data items belong much to categories he/she prefers much (C1) but *does not mind* whether the items belong to categories he/she prefers little (C2 and C3). In this case, U1 will accept D1, *D3* and reject D2. We denote this type of user as user type 1.
- Suppose U1 minds whether data items belong much to categories he/she prefers much (C1) and *also minds* whether the items belong little to categories he/she prefers little (C2 and C3). In this case, U1 will accept only D1 and reject D2, *D3*. We denote this type of user as user type 2.

To simulate such user personality, we design two methods of calculating the vector distance utilizing the product score and the Euclid distance. In the case of utilizing the product score, the distance is defined as

$$d(\vec{p},\vec{m}) = \frac{\sum_{i=1}^{c} p_i m_i}{c} \quad . \tag{2}$$

and in the case of utilizing the Euclidian distance, the distance is defined as

$$d(\vec{p},\vec{m}) = \frac{\sqrt{\sum_{i=1}^{c} (p_i - m_i)^2}}{c} \quad . \tag{3}$$

where $\vec{p}$ ={p_1, p_2, …} is the user preference vector, $\vec{m}$ ={m_1, m_2, …} is the membership vector of a data item, and c is the number of data categories.

In the case of utilizing the product score, a user accepts data items of which $d(\vec{p},\vec{m})$ is larger than a threshold value, or accepts a data item at a probability defined as a monotonically non-decreasing function of $d(\vec{p},\vec{m})$. The value of $d(\vec{p},\vec{m})$ does not increase for data categories of which the user does not prefer at all (i.e., $p_i = 0$). Thus, this variation can simulate the type 1 users.

On the other hand, in the case of utilizing the Euclidian distance, a user accepts data items of which $d(\vec{p},\vec{m})$ is smaller than a threshold value, or accepts a data item at a probability defined as a monotonically non-increasing function of $d(\vec{p},\vec{m})$. The value of $d(\vec{p},\vec{m})$ can increase for some data category c* even though a user does not prefer c* at all (i.e., p_{c*}=0): $(p_{c*} - m_{c*})^2 > 0$ if $p_{c*} \neq m_{c*}$. Thus, this variation can simulate the type 2 users.

3 Evaluation of Recommendation Ability by Simulation

We have developed a computer simulator program to evaluate the ability of P2P information recommendation based on our model. By using the program, we have tested simulations with several parameter settings. From the results of the simulations, the ability of agents in our model is evaluated. Recall and precision can be used as metrics of the ability in information recommendations [10]. The metrics are defined as follows.

$$\text{Recall} = \frac{|D_{rec} \cap D_{rel}|}{|D_{rel}|} \quad . \tag{4}$$

$$\text{Precision} = \frac{|D_{rec} \cap D_{rel}|}{|D_{rec}|} \quad . \tag{5}$$

where D_{rec} is the set of data items a user has been *rec*ommended and D_{rel} is the set of data items the user can accept if recommended (i.e., *rel*evant items).

The following shows an example of the simulation designs. The basic parameters are designed as shown in Table 3.

Table 3. Example of Basic Simulation Parameter Design

Number of users (nodes)	100
Total number of data items	100
Number of initial data items for each user	5
Number of data categories	8
Number of nodes an agent communicates for a try of recommendation	3

Each of the 100 users is randomly either the type 1 or 2 user. A preference vector for a user is designed so that p_i = 1.0 for a randomly selected category and p_i is a random value in [0, 0.3] for the other seven categories. This design simulates a situation in which each user prefers one of the eight categories very much and not the other seven categories so much.

The number of data items is also 100. A membership vector of each item is designed in the same way as a preference vector: m_i = 1.0 for a randomly selected category and m_i is a random value in [0, 0.3] for the other seven categories. This design simulates a situation in which each data item belongs to one of the eight categories very much and not the other seven categories so much. Thus, the relevant data item set D_{rel} for a user is likely to include items of which m_x = 1.0 for the category x where p_x =1.0. In this design, the 100 users and the 100 data items are likely to be categorized into eight groups.

The set D_{rel} for a user is determined as follows. The distance $d(\vec{p},\vec{m})$ is calculated 100 times for a single user and the 100 data items. If the user is type 1, $d(\vec{p},\vec{m})$ is based on the vector product so that the value $d(\vec{p},\vec{m})$ becomes *larger* as a data item meets the preference of the user more. The maximun value of $d(\vec{p},\vec{m})$ among the 100 data items are calculated (d_{max}), and D_{rel} for the user is determined as the set of data items of which $d(\vec{p},\vec{m}) \geq 0.9 * d_{max}$. In the same manner, D_{rel} for a type 2 user is determined. If the user is type 2, $d(\vec{p},\vec{m})$ is based on the Euclid distance so that the value $d(\vec{p},\vec{m})$ becomes *smaller* as a data item meets the preference of the user more. The minimum value of $d(\vec{p},\vec{m})$ among the 100 data items are calculated (d_{min}), and D_{rel} for the user is determined as the set of data items of which $d(\vec{p},\vec{m}) \leq 0.3 * d_{min}$. The threshold factors 0.9 and 0.3 are determined so that $|D_{rel}|$ becomes around 10 to 15.

Data items that each user initially has are five items randomly selected from D_{rel} of the user. In the initial state, the similarity score $S(N_a, N_b)$ between two nodes N_a and N_b is determined by these five items in N_a and N_b.

Suppose each of the 100 users receives recommendation once a specific interval of time: it is defined as one cycle that all users receive recommendation once in a random order. For a try of recommendation, an agent randomly selects three nodes in the current node list and determines data items to recommend by collaborative filtering with the three nodes.

As the recommendation probability function that characterizes agent behavior, the function in Fig 3(a) is applied. The simulation continued until no user accepted one or more items in two successive cycles.

The result of simulation with the above design is shown in Figs. 4, 5 and Table 4. Fig. 4 shows the number of users who accepted some of the recommended items and the total number of data items accepted by the users. This trial of simulation continued to 63 cycles. The maximum and mean numbers of accepting users in a cycle were 22 and 8. In average, 1.2 items were accepted per accepting user in a cycle. These values seem relatively small: this will be because, in this simulation, the 100 users (nodes) are implicitly categorized into 8+ groups and the number of nodes an agent communicates for a try of recommendation is small (three: see Table 3). Fig. 5 and Table 4 show the results of recommendation recall and precision. Fig 5(a) & Table 4(a) show the results for type 1, and Fig 5(b) & Table 4(b) show the results for type 2. A plot in Fig. 5 represents a value of recall or precision for a user (the threshold value of $d(\vec{p}, \vec{m})$ is not the same even for the users of the same type because the preference vector $\vec{p}$ is not the same).

It should be noted that D_{rel} includes data items a users initially has: these items are the relevant ones but not the recommended ones. In the calculation of recall and precision scores, these initial items are removed from D_{rel}.

Findings from the results in Fig. 5 and Table 4 are as follows.

- The agents recommended very well in terms of recall. For both type 1 and 2 users, the mean recall scores were 0.98.
- On the other hand, the agents did not recommend so well in terms of precision. In the best case the precision score was 1.0 (so that no irrelevant item was recommended to the user) but in the worst case the score was 0.11. The average precision score was 0.75 (or 0.72) for type 1 (or 2) users, which was smaller than the average recall scores.

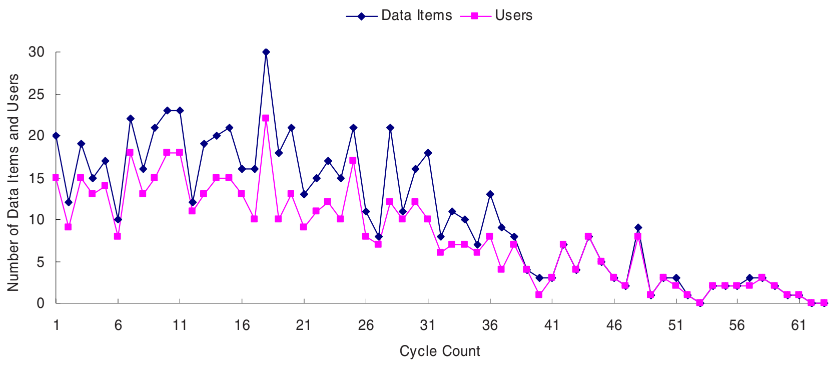
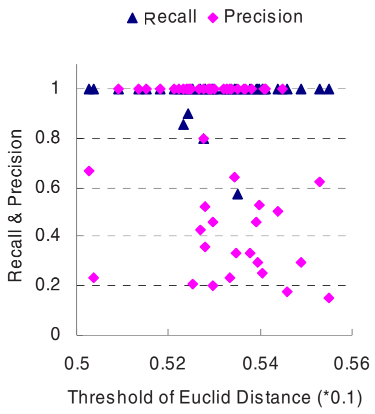
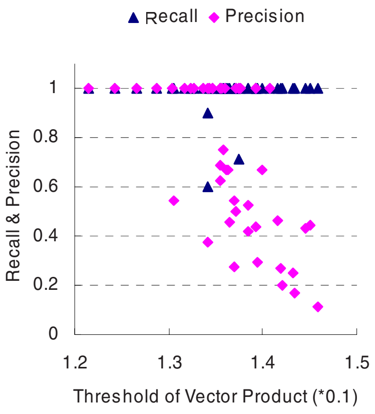


Fig. 4. Numbers of Accepting Users and Accepted Data Items

We tested additional trials of simulations in which the recommendation probability function p(S) was changed from that in Fig. 3(a) to Fig. 3(b) and Fig. 3(c), but the results were similar to the above: high scores of recall and lower scores of precision



(a) Type 1 Users



(b) Type 2 Users

Fig. 5. Recall and Precision Scores for Each User

Table 4. Statistics of Recall and Precision Scores

	ave	S.D.	max	min
Recall	0.98	0.067	1.0	0.57
Precision	0.75	0.32	1.0	0.15
F	0.80	0.25	1.0	0.26

(a) Type 1 Users

	ave	S.D.	max	min
Recall	0.98	0.071	1.0	0.60
Precision	0.72	0.30	1.0	0.11
F	0.79	0.24	1.0	0.20

(b) Type 2 Users

than recall scores. It was worse still that recall scores could be smaller in the cases (b) and (c) than in the case (a) because in the cases of (b) and (c) the recommendation probability becomes smaller.

These results indicate that agents in our P2P recommendation system model are likely to overly recommend. We find that future research should include improvements in the design of agents for better precision.

4 Conclusion

In this paper, we proposed a model of P2P information recommendation based on collaborative filtering. The model is a modified one from the model we proposed before. We have developed a computer simulator of recommendation network system based on the model. The ability of recommendation agents was evaluated by analyzing results of simulations with experimental system parameter designs. It is found that the agents are likely to overly recommend so that the recall score becomes high but the precision score becomes low. Improvement in the agent design for better precision is a research challenge in our future work.

A promising solution to the challenge is the design of agent adaptation to behaviors of their user. To make agents dynamically adapt their recommendation probability functions and the methods of selecting partner nodes for collaborative filtering to logs of their users' acceptance/rejection behaviors, precision scores are expected to improve. In addition, such adaptation is expected to enable agents follow temporal changes in users' preferences over a period of time. The user preferences are supposed as implicit in our model so that the speed with which agents follow to the changes in user preferences should be investigated.

References

1. Resnick, P., Varian, H.R.: Recommender Systems. Communications of the ACM 40(3), 56–58 (1997)
2. Riecken, D.: Introduction: Personalized Views of Personalization. Communications of the ACM 43(8), 26–28 (2000)
3. Konstan, J.A.: Introduction to Recommender Systems: Algorithms and Evaluation. ACM Transactions on Information Systems 22(1), 1–4 (2004)
4. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM 35(12), 61–70 (1992)
5. Oram, A. (ed.): Peer-to-peer: Harnessing the Benefits of a Disruptive Technology. O'Reilly and Associates (2001)
6. Lethin, R.: SPECIAL ISSUE: Technical and Social Components of Peer-to-peer Computing - Introduction. Communications of the ACM 46(2), 30–32 (2003)
7. Androutsellis-Theotokis, S., Spinellis, D.: A Survey of Peer-to-peer Content Distribution Technologies. ACM Computing Surveys 36m(4), 335–371 (2004)
8. Yanagihara, T., Iwai, M., Tokuda, H.: Designing and Implementing a Simulator for P2P Networks, Special Interest Groups on System Software and Operating System. Information Processing Society of Japan(in Japanese) 2002(60), 157–162 (2002)
9. Okada, H.: Simulation Model of P2P Information Distribution based on Collaborative Filtering. In: Proc. 11th Int. Conf. on Human-Computer Interaction (HCI International 2005), CD-ROM (2005)
10. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems 22(1), 5–53 (2004)