ACP2P: Agent-Community-based Peer-to-Peer Information Retrieval – an Evaluation

Tsunenori Mine[1], Akihiro Kogo[2], and Makoto Amamiya[1]

Department of Intelligent Systems, {Faculty[1], Graduate School[2]} of Information Science and Electrical Engineering, Kyushu University
6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580, Japan
{mine,kogo,amamiya}@al.is.kyushu-u.ac.jp,
http://www-al.is.kyushu-u.ac.jp/~mine/mine-e.html

Abstract. The Agent-Community-based Peer-to-Peer Information Retrieval (ACP2P) method[1],[2] uses agent communities to manage and look up information of interest tousers. An agent works as a delegate of its user and searches for information that the user wants by communicating with other agents. The communication between agents is carried out in a peer-to-peer computing architecture. Retrieving information relevant to a user query is performed with content files which consist of original and retrieved documents, and two histories: a query/retrieved document history and a query/sender agent history. The ACP2P is implemented using the Multi-Agent Kodama framework.
In this paper, we present some mathematical aspects of the ACP2P method with respect to the relationships between communication loads and the number of records that are stored both in the two histories and retrieved document content files, and discuss the experimental results, for which illustrate the validity of this approach. The results confirm the mathematical conjectures we presented and show that the two histories are more useful for reducing the communication load than a naive method employing 'multicast' techniques, and lead to a higher retrieval accuracy than the naive method.

1 Introduction

Although the rapid growth of the World Wide Web and the spread of the Internet have helped Internet users to access useful resources or services, users often find it difficult to search for the information they need because of the flood of information that needs to be filtered out, and lack of a clear idea of the targets they want. In order to deal with these problems, a lot of studies on information filtering (e.g. [3]), information recommendation (e.g. [4]), expert finding (e.g. [5]), and collaborative filtering (e.g. [6]) have been carried out. Most systems developed in that research are, unfortunately, based on the server-client computational model and are often distressed by the fundamental bottle-neck coming from their central control system architecture. Although some systems based on peer-to-peer (P2P for short) computing architectures have been developed and

implemented (e.g. [7], [8], [9], [10]) , each node of most of those systems only deals with simple and monolithic processing chores.

Considering these issues, we proposed an Agent Community based Peer-to-Peer information retrieval method called ACP2P method, which uses agent communities to manage and look up information related to a user query.[1],[2] The agent communities can reflect the structures of human groups or societies such as laboratories, departments, institutions, research groups and so force, where the people with the same or similar interests, objectives or aims stay together, and often browse or look for similar information from the Web. In the ACP2P method, considering such environments, an agent works as a delegate of its user and searches for information that the user wants by communicating with other agents. The communication between agents is carried out based on a P2P computing architecture. In order to retrieve information relevant to a user query, an agent uses two histories: a query/retrieved document history (Q/RDH for short) and a query/sender agent history (Q/SAH for short). The former is a list of pairs of a query and retrieved document information, where the queries were sent by the agent itself and the document information includes the addresses of both agents that returned the document and those that created or owned the document. The latter is a list of pairs of a query and a sender agent's address and shows "who sent what query to the agent." This is useful for finding new information sources. Making use of the Q/SAH is expected to have a collaborative filtering effect, which gradually creates virtual agent communities, where agents with the same interests stay together. We have demonstrated through several experiments that the method reduced communication loads much more than other methods which do not employ Q/SAH to look up a target agent, and was useful for creating a "give and take" effect, i.e., as an agent receives more queries, it acquires more links to new knowledge[11], but have not so far discussed any mathematical aspects of the method or the retrieval accuracy of the method.

In this paper, we present some mathematical aspects of the ACP2P method with respect to the relationships between communication loads and the number of records that are stored both in the two histories and retrieved document content files, and discuss the experimental results to illustrate the validity of this approach. The results confirm our mathematical conjectures about the ACP2P method and show that two histories are more useful for reducing communication loads than a naive method employing 'multicast' techniques, and lead to a higher retrieval accuracy than the naive method. The remainder of the paper is structured as follows. Section 2 considers the ACP2P method. Section 3 discusses the experimental results and Section 4 describes related work.

2 ACP2P Method

2.1 Overview of the ACP2P Method Implemented with Multi-Agent Kodama

The ACP2P method employs three types of agents: user interface (UI) agent, information retrieval (IR) agent and history management (HM) agent. A set of

three agents (UI agent, IR agent, HM agent) is assigned to each user. Although a UI agent and an HM agent communicate only with the IR agent of their user, an IR agent communicates with other users' IR agents not only in the community it belongs to, but also in other communities, to search for information relevant to its user's query. A pair of Q/RDH and Q/SAH histories and retrieved content files are managed by the HM agent.

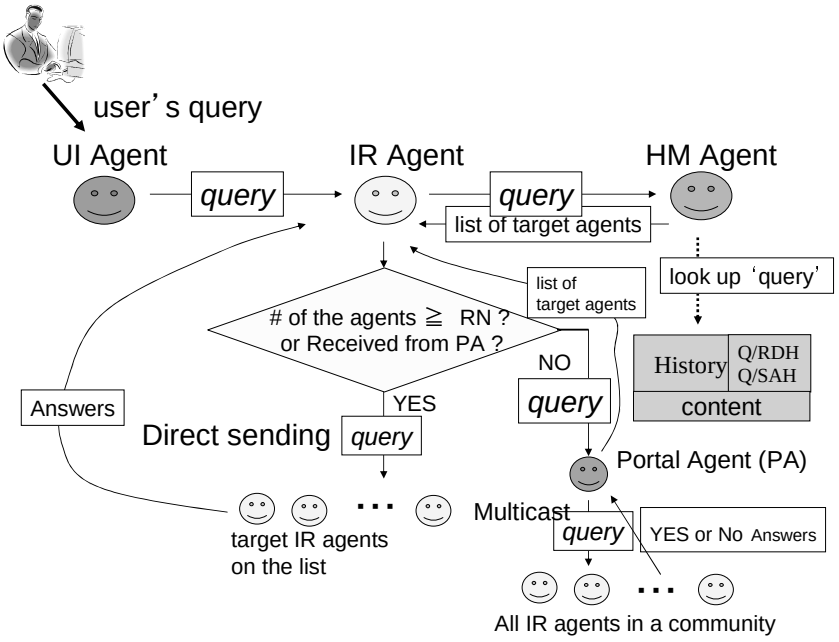

Fig. 1. Actions for Sending a Query

When receiving a query from a UI agent, an IR agent asks an HM agent to look up target agents with its history or asks a portal agent to do it using a query multicasting technique. (Fig.1).

When receiving a query from other IR agents, an IR agent looks up the information relevant to the query from its original content and retrieved content files, sends an answer to the query-sender IR agent, and also sends a pair of the query and the address of the query-sender IR agent to an HM agent so that it can update Q/SAH (Fig.2 (left)).

The returned answer is either a pair of a 'Yes' message and retrieved documents or a 'No' message indicating that there is no relevant information, although retrieved documents are not returned when the query comes through a portal agent. When receiving answers with a 'Yes' message from other IR agents,

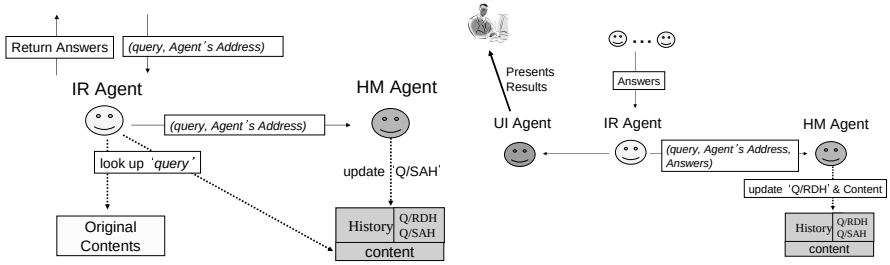

Fig. 2. Actions for Receiving a Query (left) and for Receiving Answers (right)

the IR agent sends them to a UI agent, and sends them with a pair of a query and the addresses of answer sender IR agents to an HM agent (Fig.2 (right)).

The ACP2P method is implemented with Multi-Agent Kodama (Kyushu university Open & Distributed Autonomous Multi-Agent) [12]. Kodama comprises hierarchical structured agent communities based on a portal-agent model. A portal agent is the representative of all member agents in a community and allows the community to be treated as one normal agent outside the community. A portal agent has its role limited in a community, and the portal agent itself may be managed by another higher-level portal agent. A portal agent manages all member agents in its community and can multicast a message to them. Any member agent in a community can ask the portal agent to multicast its message. The portal agent has its role limited in a community, and itself may be managed by another higher-level portal agent.

Fig.3 shows an example of the agent community structure which the ACP2P method is based on. A portal agent in the figure manages all member agents' addresses there, where a member agent of a community designates an IR agent. When a member agent wants to find any target agents which have information relevant to a query, the agent looks them up using two histories: Q/RDH and Q/SAH, and Content files. If the target agents are found, a query is sent directly to them, and their retrieved results are also returned directly to the query-sender IR agent. If the requested number (N_R) of such agents is not found, the agent asks the portal agent to send the query to all the other member agents in the community by a query multicasting technique. At that time, all the answers will be returned to the portal agent. If the number of results with a 'Yes' message reaches N_R, without waiting for the rest of answers from other IR agents, the portal agent sends them back to the query-sender IR agent. Even if the number of 'Yes' messages did not reach N_R after all other IR agents replied, the portal agent still sends the currently held results to the query-sender IR agent.

2.2 Communication Load and History Size

MultiCast: Without Using Two Hisotries In the ACP2P method, every IR agent sends one query in rotation. Since an IR agent initially has no records

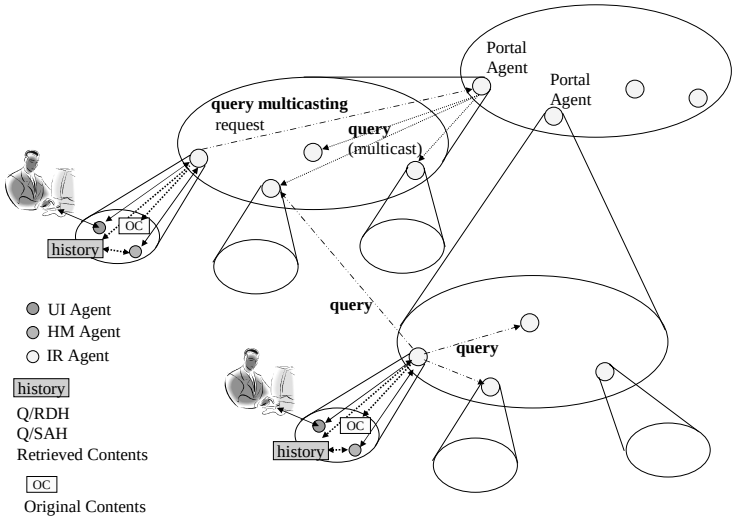

Fig. 3. Agents and their Community Structure

in its histories, the IR agent first has to ask a portal agent to multicast the query to all other IR agents in its community. After receiving the query, the IR agents return a 'Yes' or 'No' message with their address to the portal agent. Then the portal agent selects the top N_R IR agents returning 'Yes' messages in the order they are received, makes a list of them and sends the list to the query-sender IR agent without waiting for the rest of other IR agents' answers. After receiving the list, the query-sender IR agent again sends the query to the IR agents on the list. These processes of exchanging messages are shown as follows:

1. QS-IRA $-(1)\rightarrow$ PA $-(N-1)\rightarrow$ A-IRAs
2. QS-IRA $\leftarrow(1)-$ PA $\leftarrow(N-1)-$ A-IRAs
3. QS-IRA $-(N_R)\rightarrow$ T-IRAs
4. QS-IRA $\leftarrow(N_R)-$ T-IRAs

Where QS-IRA, PA, A-IRA and T-IRA represent a query-sender IR agent, a portal agent, all other IR agents, and target IR agents, respectively. The number in the parentheses on the arrow represents the number of messages received by the agent (or agents) pointed by the arrow. N is the number of IR agents in a community. During the period when every IR agent sends one query in rotation, the total number of messages exchanged among all IR agents and a portal agent is at most $2(N + N_R)N$, the number of messages received by a portal agent is at most N^2 and the average number of messages received by an IR agent is $2N_R + N$. Therefore, when the multicast technique is employed, the number of messages received by each IR agent in one routine is proportional to N because $N >> N_R$.

Using Two Histories As more queries are sent, more records will be accumulated in the two histories of an IR agent. Let N_{RD} be the maximum number of documents returned by target IR agents that received a query. At that time, the maximum number of documents to be stored in a Content file holding retrieved documents (# in Content for short) will be $N_R \times N_{RD}$, and the maximum number of pairs in Q/RDH of a query and the address of an IR agent that replied to the query (# in Q/RDH for short) will be N_R. Since the number of these records to be stored in a Content file or Q/RDH is proportional to the number of queries to be sent by an IR agent, after the IR agent sends N_Q queries, # in Content and # in Q/RDH will be $N_Q \times N_R \times N_{RD}$ and $N_Q \times N_R$, respectively. An IR agent receives at most $(N-1) \times N_Q$ queries when an IR agent happens to receive a query from all other IR agents. Then, the Q/SAH will hold $(N-1) \times N_Q$ records, which are pairs of a query and a query-sender IR agent's address. On the average, Q/SAH will hold N_Q records.

When an IR agent sends a query, it searches for N_R target IR agents from both its retrieved document content file and the two histories. When N_R target candidate IR agents or more were found, the query-sender IR agent ranks the IR agents based on the similarity of the query and selects the top N_R IR agents from among them. The similarity measure will be described in Sec. 3.3. Otherwise, the query-sender IR agent has to ask a portal agent to multicast the query to all the other IR agents so that the IR agent can fulfill its quota of target IR agents. For every query sending of each IR agent, $2N_R$ messages will be exchanged in the former case, and $2N_R + 2N$ messages in the latter case, as mentioned earlier. As more queries are sent by IR agents, the number of occurrences of the latter case, i.e. multicasting, will be reduced according to the increase in records in their Content files and histories.

3 Experiments

3.1 Preliminaries

We used the Web pages of Yahoo! JAPAN [13] for the experiments as Mine et al. [11] did. The Web pages used are broadly divided into five categories: animals, sports, computers, medicine, and finance. Each of them consists of 20 smaller categories, which are selected in descending order of the number of Web pages recorded in a category. An IR agent is assigned to each selected category, and thus 100 IR agents are created and activated in the experiments. A category name is used as the name of an IR agent, and the Web pages in the category are used as the original documents of the agent. Experiments are conducted with 4 PCs connected to Gigabit Ethernet. 25 IR agents are assigned to each PC, but all 100 IR agents are assigned to a single community for simplicity.

We conducted experiments to show how the two histories help to reduce communication loads between agents looking for information relevant to a query and how Q/SAH helps in searching for new information sources, to having a higher accuracy in retrieving documents or one comparable to a method without two histories. To perform the experiments, we compared three methods : (1)

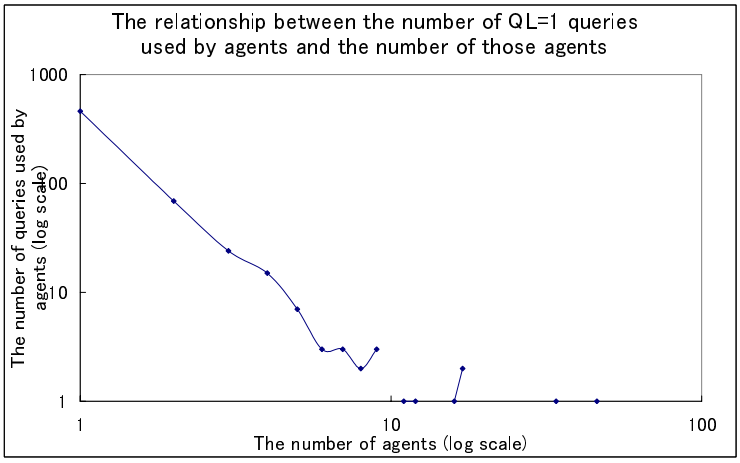

Fig. 4. The relationship between the number of QL=1 queries used by agents and the number of those agents. Both axes are in log scale.

ACP2P with a Q/SAH (wQ/SAH for short), (2) ACP2P without a Q/SAH (woQ/SAH for short), and (3) a simple method always employing a 'multicast' technique (MulCst for short).

In the experiments, two query sets : QL=1 and QL=2, were used. QL=1 and QL=2 consist of 10 queries, whose query length is one and two, respectively, where query length means the number of terms in a query. When using queries belonging to QL=1, 10 nouns are extracted from every category assigned to each IR agent in descending order of their frequency of occurrence in the category. Each noun is used as a query of the IR agent. When using those belonging to QL=2, 5 nouns are extracted and the combinations of the extracted 5 nouns taken in pairs create 10 queries.

The relationship between the number of QL=1 queries used by agents and the number of such agents is shown in Fig.4, where both axes are in log scale. The relationship almost seems to obey a power law distribution.

3.2 Relevance Judgement and Evaluation

In a P2P network environment, gathering all documents from every peer is not always possible, that is, indexing all documents is quite difficult. Thus the first goal for IR in the P2P network environment is to achieve a result comparable with a conventional IR method (CIR method for short). As the CIR method, we employed a Probabilistic IR method that applies a simplified BM25 [14] weighting function to all the documents collected from every peer. The simplified BM25 is defined as follows:

$$\sum_{T \in Q} \log \frac{n + 0.5}{N - n + 0.5} \frac{2tf}{\frac{dl}{avdl} + tf} \tag{1}$$

Where Q is a query that contains terms T. tf is the frequency of occurrence of the term within a specific document. N and n are the number of items (documents) in the collection[1] and the number of documents containing the term, respectively. dl and $avdl$ are respectively the document length and average document length, where the document length is the number of terms in a document, and a term is a word detected by a morphological analyzer.

In order to compare the ACP2P method with the CIR method, we used the following equation:

$$\sum_{i=1}^{N_R} \frac{1}{r(i)} / \sum_{i=1}^{N_R} \frac{1}{i}$$

Where $r(i)$ is the CIR method's rank of the document that is ranked by the ACP2P as the ith document. For example, if a document is ranked by the ACP2P as the 2nd document and the document's rank by the CIR method is 3, then this means that $r(2)$ returns 3. We call this measure *Reciprocal Rank Similarity* (RRS for short). We can assume that RRS's denominator $\sum_{i=1}^{N_R} \frac{1}{i}$ represents the ideal value of a given model, where it is the CIR method in this paper. As the ACP2P approaches the given model, the RRS value becomes higher. Thus, the RRS can measure the similarity between ranks generated by the ACP2P and by the CIR and returns a higher score the smaller $r(i)$ $(1 \leq i \leq N_R)$ is, i.e., the higher the rank. For example, if a user wants to find 3 documents relevant to his/her query and we suppose the top 3 ranked documents' rank returned by his/her agent to be 3, 5 and 1, then the RSS returns $\frac{1/3+1/5+1/1}{1/1+1/2+1/3} = 0.836$.

In the experiment, we use an average RRS: $\frac{1}{N_a} \sum_{i}^{N_a} RRS(i)$, where N_a is the the number of all IR agents and $RRS(i)$ is the RRS of the ith IR agent.

3.3 Similarity Measure for Detecting Target Agent

In order to find N_R target agents to be sent a query, we calculate $Score(query, t\_agent)$, which returns the similarity value between query $query$ and target agent t_agent, with equation (2); $Score(query, t\_agent)$ becomes higher if t_agent sends a greater number of similar queries and returns more documents related to $query$.

$$\begin{aligned} Score(query, t\_agent) = & \sum_{i=1}^{k} \cos(\boldsymbol{query}, \boldsymbol{qh_{di}}) \\ & + \sum_{i=1}^{m} (\cos(\boldsymbol{query}, \boldsymbol{qh_{sai}}) + \varphi(i)) \\ & + max_{1 \leq i \leq n} Sim_d(\boldsymbol{query}, \boldsymbol{doc_i}) \end{aligned} \tag{2}$$

$$\varphi(i) = \begin{cases} \delta \text{ if } qh_{sai} \text{ is a query directly sent by} \\ \quad \text{an other IR agent.} \\ 0 \text{ otherwise} \end{cases}$$

[1] In the experiment, the documents in the collection are those collected from all peers.

In equation (2), *query* consists of $w_1, ..., w_m$, and w_i $(1 \leq i \leq m)$ is a term in *query*. $\boldsymbol{query}$ is the term vector whose element is the frequency of occurrence of the term in *query*. qh_d and qh_{sa} represent a query in a record of Q/RDH and Q/SAH, respectively. The first term $\sum_{i=1}^{k} \cos(\boldsymbol{query}, \boldsymbol{qh_{di}})$ returns the total score of the similarities between *query* and each of k number of queries sent to *t_agent*. The second term $\sum_{i=1}^{m}(\cos(\boldsymbol{query}, \boldsymbol{qh_{sai}}) + \varphi(i))$ represents the score between *query* and qh_{sa_i}, which is the *i_th* of m queries sent by *t_agent* in Q/SAH. $\varphi(i)$ is a weight to consider the importance of 'direct sending of a query'. If qh_{sa_i} is sent directly by *t_agent*, δ is added to the score. In order to decide the value of φ, we performed a simple pre-experiment that compared $\varphi = 0$ with $\varphi = 0.1$. Since the result of $\varphi = 0.1$ was better than that of $\varphi = 0$, we employed $\varphi = 0.1$. The last term $max_{1 \leq i \leq n} Sim_d(\boldsymbol{query}, \boldsymbol{doc_i})$ is the maximum score of similarity between *query* and each of n documents originally created by the user of *t_agent* or just returned by *t_agent*. $Sim_d(query, doc)$ represents the similarity between *query* and the content of retrieved document *doc*. It is calculated with a more simplified of version of BM25, in which $\frac{dl}{avdl}$ in equation (1) is set to 1.

After calculating $Score(query, t\_agent)$ for each IR agent *t_agent* in the retrieved document Content file and two histories: Q/RDH and Q/SAH, N_R target agents will be selected in the descending order of $Score(query, t\_agent)$, which should be greater than 0. Whenever N_R agents are not found, a query-sender IR agent asks a portal agent to multicast a query to all the other IR agents. If a target IR agent finds information relevant to *query* from its original or retrieved document content files with Sim_d in equation (2), it returns a 'Yes' message, otherwise a 'No' message. If the similarity value between a document and a query that was returned by Sim_d is greater than some threshold value (0 for the experiments), the document will be scvbnjudged relevant, otherwise irrelevant.

3.4 Experimental Results

First we compare the change of the average number of messages exchanged by each IR agent for every query input. For the comparison, we use 3 different request numbers: N_R=3, 10 and 20. The results are shown in Fig.5. In the figure the vertical axis is the average number of messages and the horizontal axis is the number of queries sent by each IR agent. The left side in the figure shows the result of using QL=1 and the right side shows that of using QL=2.

The results show that the average number of messages received by each IR agent is reduced for every query input. In particular, when using QL=2, the number of received messages decreases more quickly than for QL=1, and almost converges at the third query input because there is a larger number of identical words in QL=2 queries than those of QL=1, and consequently the words in QL=2 queries are more frequently found in two histories and retrieved document files than is the case for QL=1. Furthermore we can see that the graph of the number of messages in woQ/SAH approaches more closely to that in MulCst than that for wQ/SAH, as N_R increases. Thus we can say that Q/SAH history is quite

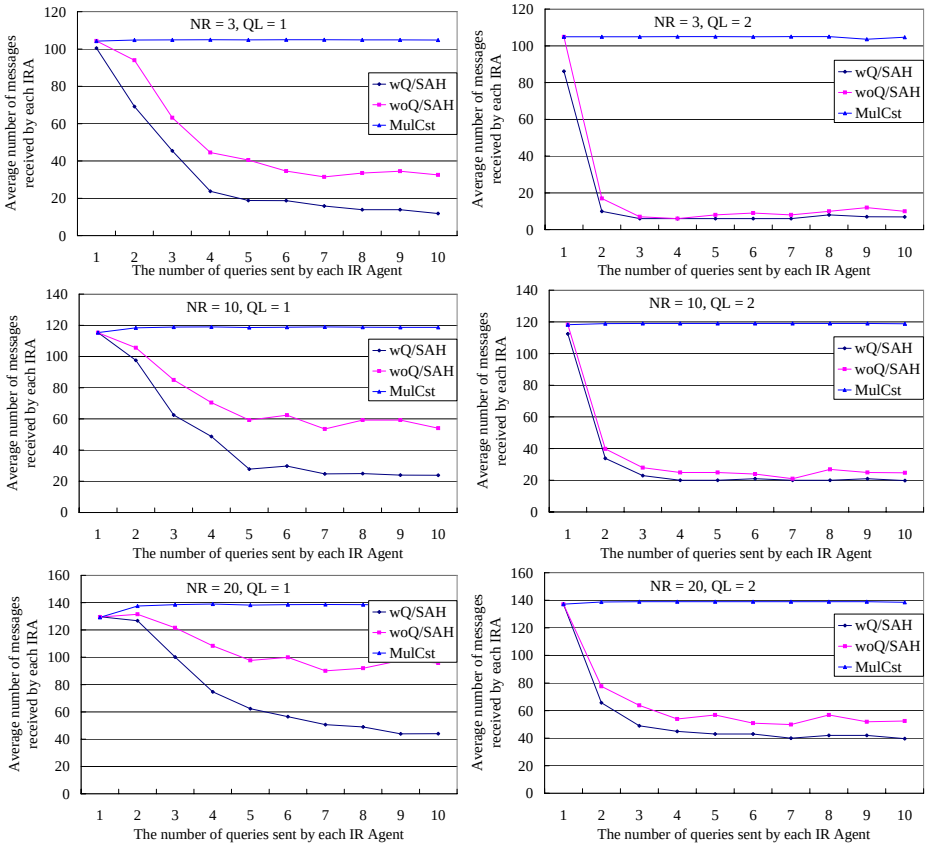

Fig. 5. The comparison of average number of messages received by each IR agent for every query input using 3 diifferent N_R values: N_R=3 (TOP), N_R=10 (MID) and N_R=20 (BTM). The query belongs to either QL=1 (left) or QL=2 (right).

useful for finding target agents related to queries, in particular when a user uses a greater number of different queries which include fewer identical words.

Next we compare the RRS values of the three methods under the same conditions as in the previous experiment. The results are shown in Fig.6. In the figure the vertical axis is the average RRS and the horizontal axis is the number of queries sent by each IR agent.

As the value of N_R increases, the RRS value also increases and the curve of the graphs becomes flatter. We can see that the RRS value of MulCst increases as the number of queries sent increases. Considering this phenomenon, we surmise that original documents assigned to IR agents will gradually be spread over the community through the document retrieval process of each IR agent. Thus even though a portal agent selects target agents in the order their 'Yes' messages are received, the probability that higher weighted documents will be returned

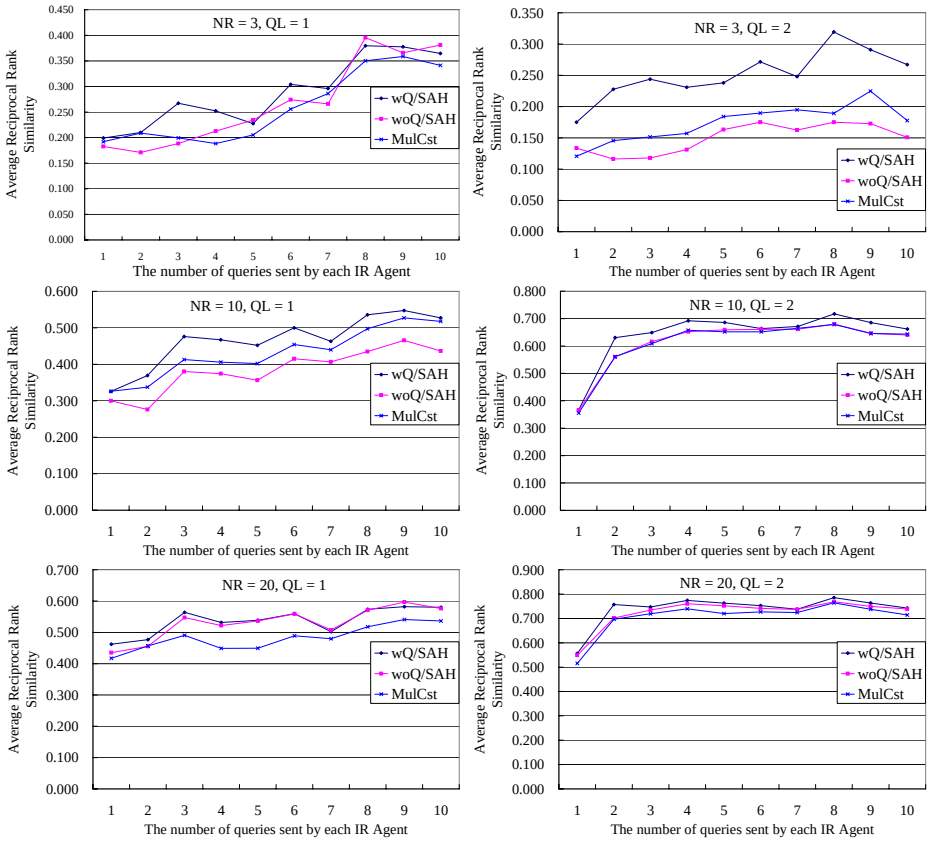

Fig. 6. The comparison of average reciprocal rank similarity (RRS) of each IR agent for every query input using 3 diifferent N_R values : N_R=3 (TOP), N_R=10 (MID) and N_R=20 (BTM). The query belongs to either QL=1 (left) or QL=2 (right).

rises. For the same reason, since the RRS value of the MulCst increases as N_R increases, the difference of the three methods decreases.

When using QL=1, the wQ/SAH almost achieves higher retrieval accuracy than the other two methods, although the RRS value is unfortunately not so high because the records stored in the Content files and the two histories are originally acquired by a portal agent using the query multicasting technique and its RRS value is not so high. However, when using QL=2, all three methods identically achieve high RRS scores at both N_R=10 and 20.

Lastly, we check the cases where the number of documents returned by each target IR agent is limited to N_R, and the k, m, and n value of $Score(query, t\_agent)$ is also limited to N_R so as to select the N_R highest values of each term, respectively. Although we will not be able to show the details here due to space limitations, the graphs of the results achieved by this experiment are similar to the ones shown in Fig.5 and Fig.6.

4 Related Work

There is lots of work related to the topics considered in this paper, such as P2P file searching, Multi-Agent based Information Retrieval and so forth. Freenet [8] and Chord [7] are carried out in a pure P2P computing architecture. They neither employ 'broadcast' techniques like Gnutella [9], nor have a centralized server machine like Napster[10]. Freenet provides information-sharing and information-finding functions among anonymously distributed nodes. Although Chord does not provide anonymity of nodes, it has an efficient protocol for looking up nodes with Distributed Hash Tables. Their node searching strategies are conducted according to keywords attached to the information of the nodes. Thus users need to know the keywords of the information they want to search for. On the other hand, since the ACP2P method can make use of the content information of documents and two histories; Q/RDH and Q/SAH, it allows the users to perform a more flexible search for target agents with relevant information.

Routing Indices (RIs)[16] are local routing indices that nodes use so that they can forward queries to neighbors that are more likely to have answers. If a node cannot answer a query, it forwards the query to a subset of its neighbors based on its local RI. The RI stores information concerning which neighbors have what topics of documents, and thus gives a "direction" towards the document, rather than its actual location[16]. On the other hand, the ACP2P method directly searches for target agents with relevant information, using retrieved documents and two histories. In particular, Q/SAH provides similar effects to link analysis like the PageRank[17] or HITs algorithm[18], and can be expected to make a natural collaborative filtering effect emerge because users want to send a query again to the peers that can return results which satisfy them, and vice versa.

I-Gaia [19] is an application layer for information processing in the DIET architecture, which is a Multi-Agent System development platform. ACP2P is also a Multi-Agent-based application, but it does not use a mediator agent like t-infocytes of I-Gaia to learn appropriate paths between agents in sending queries or publishing documents. NeuroGrid[20] is an adaptive decentralized search system which supports distributed search by forwarding queries based on the contents of each network node, and supports a learning mechanism that dynamically adjusts metadata describing the contents of nodes and the files that make up those contents, using users' positive and negative feedback. However, there is no discussion of the accuracy of NeuroGrid, and it does not use a history like Q/SAH of ACP2P.

5 Conclusions and Future Work

We presented some mathematical aspects of the ACP2P method and discussed the experimental results that illustrate its validity. To do the experiments, we implemented the method with Multi-Agent System Kodama.

We conducted several experiments to show whether or not two histories helped to reduce communication loads between agents in searching for information relevant to a query, and whether or not Q/SAH helped in looking up

new information sources. The experimental results showed that the two histories are quite useful for looking up new information source and for reducing communication loads, and have a higher accuracy in retrieving documents than a simple method employing a multicast technique. Although the RRS value of the ACP2P method (wQ/SAH and woQ/SAH) for the CIR method using a simplified BM25 was not so high, it is because the records stored in the Content files and two histories were originally based on results selected in the order they were received by a portal agent. Therefore if we improve the way a portal agent selects the results, for example, make it wait for a greater number of results than N_R and select N_R of them in order of their weighting score, we will probably be able to achieve a higher similarity, although this method might require more time than the current method. Another method is to employ query routing before asking a portal agent to multicast the query, that is, to let an IR agent ask target IR agents to forward a query to the IR agents which are relevant to the query and are stored in the two histories of the target IR agents.

We are currently continuing experiments to achieve results with more than one hierarchical agent community, and with dynamic community environments which agents freely join and leave, and where agents update their contents so that we can simulate more realistic environments and evaluate the scalability of the ACP2P method. Those experiments will be conducted on PC clusters with 32 nodes connected to Giga-bit Ethernet. Furthermore, we are investigating how we can make use of user feedback embedded into the results in order to reflect it in ranking of retrieved documents to achieve a higher retrieval accuracy according to some measure specific to the user. We will report these results in the near future.

Acknowledgment

This research was partly supported by the Japan Society for the Promotion of Science under the Grant-in-Aid for Scientific Research (C) No. 16500082.

References

1. Mine, T., Matsuno, D., Takaki, K., Amamiya, M.: Agent community based peer-to-peer information retrieval. In: Proc. of Third Int. Joint Conf. on Autonomous Agents and Multi Agent Systems (AAMAS 2004). (2004) 1484–1485 poster.
2. Mine, T., Matsuno, D., Kogo, A., Amamiya, M.: Acp2p : Agent community based peer-to-peer information retrieval. In: Proc. of Third Int. Workshop on Agents and Peer-to-Peer Computing (AP2PC 2004). (2004) 50–61 full paper.
3. Lang, K.: NewsWeeder: learning to filter netnews. In: Proceedings of the 12th International Conference on Machine Learning, Morgan Kaufmann publishers Inc.: San Mateo, CA, USA (1995) 331–339
4. Schafer, J.B., Konstan, J.A., Riedi, J.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM Conference on Electronic Commerce. (1999) 158–166
5. Yimam-Seid, D., Kobsa, A.: Expert finding systems for organizations: Problem and domain analysis and the demoir approach. Journal of Organizational Computing and Electronic Commerce **13** (2003) 1–24

6. Good, N., Schafer, J.B., Konstan, J.A., Borchers, A., Sarwar, B.M., Herlocker, J.L., Riedl, J.: Combining collaborative filtering with personal agents for better recommendations. In: AAAI/IAAI. (1999) 439–446
7. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications. (2001) 149–160
8. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A distributed anonymous information storage and retrieval system. Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, http://www.doc.ic.ac.uk/~twh1/academic/ (2001)
9. Gnutella: http://gnutella.wego.com/ (2000)
10. Napster: http://www.napster.com/ (2000)
11. Mine, T., Matsuno, D., Kogo, A., Amamiya, M.: Design and implementation of agent community based peer-to-peer information retrieval method. In: Proc. of Eighth Int. Workshop CIA-2004 on Cooperative Information Agents (CIA 2004), LNAI 3191. (2004) 31–46
12. Zhong, G., Amamiya, S., Takahashi, K., Mine, T., Amamiya, M.: The design and application of kodama system. IEICE Transactions INF.& SYST. **E85-D** (2002) 637–646
13. Yahoo: http://www.yahoo.co.jp/ (2003)
14. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.: Okapi/keenbow at trec-8. In: NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC-8). (1999) 151–162
15. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.: Okapi at trec-3. In: NIST Special Publication 500-226: The Third Text REtrieval Conference (TREC-3). (1994)
16. Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. In: the 28th International Conference on Distributed Computing Systems. (2002)
17. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: Proc. of 7th International World Wide Web Conference:WWW7 Conference. (1998)
18. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM **46** (1999) 604–632
19. Gallardo-Antolin, A., Navia-Vasquez, A., Molina-Bulla, H.Y., Rodriguez-Gonzalez, A.B., Valverde-Albacete, F.J., Cid-Sueiero, J., Figueras-Vidal, A., Koutris, T., Xiruhaki, C., Koubarakis, M.: I-Gaia : an Information Processing Layer for the DIET Platform. In: the first international joint conference on Autonomous Agents and Multi Agent Systems (AAMAS). (2002) 1272–1279
20. Joseph, S.: Neurogrid: Semantically routing queries in peer-to-peer networks. In: the International Workshop on Peer-to-Peer Computing (co-located with Networking 2002),http://www.neurogrid.net/php/publications.php. (2002)