ACP2P : Agent Community based Peer-to-Peer Information Retrieval

Tsunenori Mine, Daisuke Matsuno, Akihiro Kogo, and Makoto Amamiya

Dept. of Intelligent Systems, Kyushu University
6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580, Japan
{mine,kogo,amamiya}@al.is.kyushu-u.ac.jp,
WWW home page: http://www-al.is.kyushu-u.ac.jp/~mine

Abstract. This paper proposes an agent community based information retrieval method, which uses agent communities to manage and look up information related to users. 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.
In order to retrieve relevant information to a user query, an agent uses two histories : a query/retrieved document history(Q/RDH) and a query/sender agent history(Q/SAH). The former is a list of pairs of query and retrieved document information, where the queries were sent by the agent itself. The latter is a list of pairs of query and sender agents and shows "who sent what query to the agent". This is useful to find a new information source. Making use of the Q/SAH is expected to cause a collaborative filtering effect, which gradually creates virtual agent communities, where agents with the same interests stay together. Our hypothesis is that a virtual agent community reduces communication loads to perform a search. As an agent receives more queries, then more links to new knowledge are achieved. From this behavior, a "give and take"(or positive feedback) effect for agents seems to emerge.
We implemented this method with Multi-Agents Kodama, and conducted the experiments to test the hypothesis. The empirical results showed that the method was much more efficient than a naive method employing 'multicast' techniques only to look up a target agent.

1 Introduction

The rapid increase of World Wide Web has made conventional search engines suffer from decreasing coverage of searching the Web. The Internet users meet information floods everyday that are forced to filter out and choose the information they need.

In order to deal with these problems, a lot of studies on distributed information retrieval(e.g. [3]), information filtering(e.g. [11]), information recommendation (e.g. [17]), expert finding(e.g. [21],[9]), or collaborative filtering (e.g. [6],[15],[7],[16]) have been carried out. Most systems developed on those researches are, unfortunately, based on the server-client computational model and

are often distressed by the fundamental bottle neck coming from its central control system architecture. Although some systems based on the peer-to-peer (P2P for short) computing architecture (e.g. [19],[4],[5],[14]) have been developed and implemented, each node of most those systems only deals with simple and monolithic processing things.

Considering these issues, we presents an Agent Community based P2P information retrieval method (ACP2P method for short), which uses agent communities to manage and look up information related to a user query. 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 related 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 query and retrieved document information, where the queries were sent by the agent itself and the document information includes the addresses of agents that replied the document. The latter is a list of pairs of query and sender agents and shows "who sent what query to the agent". This is useful to find a new information source. Making use of the Q/SAH is expected to cause a collaborative filtering effect, which gradually creates virtual agent communities, where agents with the same interests stay together. Our hypothesis is that a virtual agent community reduces communication loads to perform a search. As an agent receives more queries, then more links to new knowledge are achieved. From this behavior, a "give and take"(or positive feedback) effect for agents seems to emerge. We conducted the experiments to test the hypothesis, i.e., to evaluate how much the Q/SAH work for reducing communication loads and for causing a "give and take" effect. The experimental results showed that the method was much more efficient than the other methods without employing history Q/SAH to look up a target agent, and was useful to cause a "give and take" effect.

The remainder of the paper is structured as follows. Section 2 shows the ACP2P method. Section 3 discusses the experimental results and Section 4 describes the related work.

2 Agent Community based Peer-to-Peer Information Retrieval Method

2.1 Overview of the ACP2P Method

The ACP2P method employs the three-type agents: user interface(UI) agent, information retrieval(IR) agent and history management(HM) agent. A set of the three-type agents (UI agent, IR agent, HM agent) is assigned to each user. Although a UI agent and a HM agent communicate only with the IR agent of their user, an IR agent however can communicate with other user's IR agents not only in the community it belongs to, but also in other communities, to search for relevant information to its user's query. A pair of Q/RDH and Q/SAH is

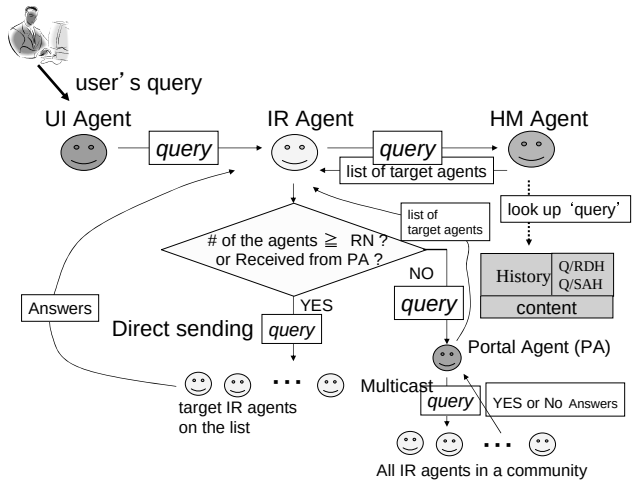

Fig. 1. Actions for Sending a Query

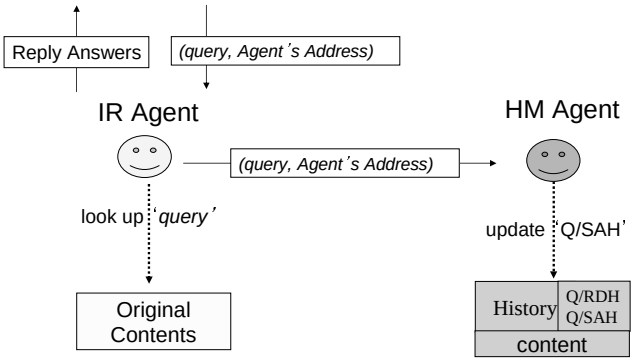

Fig. 2. Actions for Receiving a Query

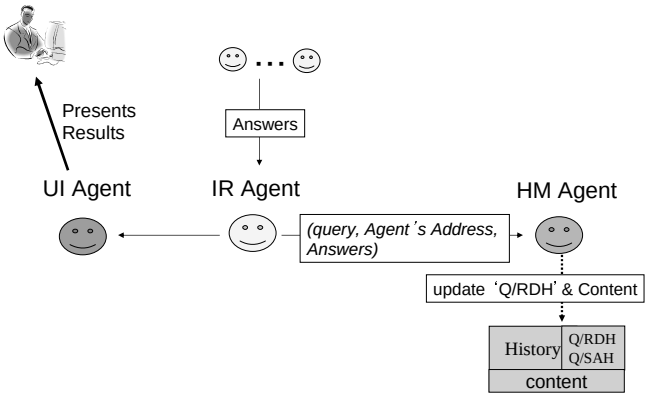

Fig. 3. Actions for Receiving Answers

managed by the HM agent. Figure 1, 2 and 3 show the processes or data flows in the cases that an IR agent sends a query, an IR agent receives a query from an other IR agent or a portal agent, and an IR agent receives answers from other IR agents, respectively. When receiving a query from a UI agent, an IR agent asks a HM agent or a portal agent to look up target agents with its history or using a query multicasting technique, respectively(Fig. 1). When receiving it from other IR agent, an IR agent looks up the relevant information to a query, sends an answer to the query sender IR agent, and sends a pair of query and the address of the query sender IR agent to a HM agent so that it can update Q/SAH (Fig. 2). The replied answer consists of a pair of a 'YES' message and retrieved documents or a 'No' message representing no relevant information, provided that retrieved documents are not replied in case the query is sent via a portal agent by a multicast technique. When receiving answers with a 'YES' message from other IR agents, an IR agent sends them to a UI agent, and sends them with a pair of query and the addresses of answer sender IR agents to the a HM agent (Fig. 3). Figure 4 shows an example of agent community structure which the ACP2P method is based on. A portal agent in the figure is the agent which is a

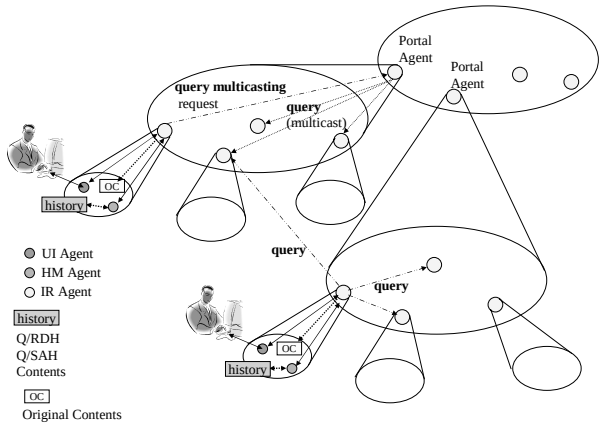

Fig. 4. Agents and their Community Structure

representative of a community and manages all member agent 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 relevant information to a query, the agent looks up them using both histories: Q/RDH and Q/SAH, and Content files. If the target agents were found, a query is directly sent to them, and their retrieved results are also directly replied to the query sender IR agent. If the request number of such agents were not found, the agent asks the portal agent to send the query to the all member agents in the community by a multicast technique. At that time, all the answers will be replied to the portal agent. If the number of the results with a 'YES' message reaches the request number, without waiting for the rest of answers by other IR agents, the portal agent sends them back to the query sender IR agent. Even if the number of 'YES' messages did

Table 1. The structures of Content file, Q/RDH file, and Q/SAH file

Content	title	the title of document
	text	the content of document
	original	the address of the IR agent whose user created the document
	range	the range allowed to be distributed(ALL, Community, Agent)
Q/RDH	query	queries sent by the agent recorded in the from field
	from	the address of IR agent replied to the query in the query field
Q/SAH	query	queries sent by the agent recorded in the from field
	from	the address of the IR agent who sent the query in the query field

not reach the request number after all IR agents replied, the portal agent also sends the currently keeping results to the query sender IR agent.

2.2 Document Content and Histories

Table 1 shows the formats of a retrieved document file: **Content** and two histories: **Q/RDH** and **Q/SAH**. The document content file consists of a list of 4-tuples <**title, text, original, range**>, each of which is the title of a retrieved document, its text content, the address of IR agent whose user owns the document, and the range allowed the document to be distributed, respectively. Since the same contents originally created by the same user sometimes happen to be replied by two or more IR agents, they are shared into the Content file.

Table 2. A part of document content

title	text	original	range
Netscape informal FAQ Japanese version	HTML text in the file	com_Netscape@	ALL

Table 3. A part of Q/SAH

query	from
telegram	root.p2p.com_telegram@
treatment	root.p2p.sic_hepatitis_type_C@
Asthma	root.p2p.sic_Asthma@
Human	root.p2p.sic_Adult_Children@
Thing	root.p2p.sic_Alzheimer's_Disease@
Ill	root.p2p.sic_Jacob_Disease@
Dream	root.p2p.sic_Dealer@
Mastocarcinoma	root.p2p.sic_Mastocarcinoma@
Hoof	root.p2p.sic_Hoot-and-Mouth-Disease@

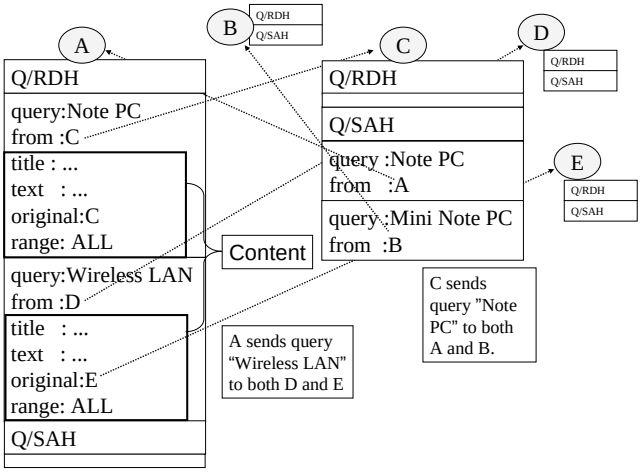

Fig. 5. Example to find target IR agents using two histories. A, B, C, D and E in circles represent IR agents' name, respectively.

The Q/RDH file comprises a list of pairs of <**query, from**>, each of which is a query sent by the agent itself and the address of IR agent that replied this retrieved information, respectively. The Q/SAH file is a list of pairs <**query, from**>, each of which is a query and the address of the agent which sent the query to the IR agent. Table 2 shows an example of part of a document content file. Table 3 also shows an example of part of Q/SAH file, which was originally written in Japanese, but translated in English.

2.3 Determining Target Agents using both Histories

In order to determine the target agents to send a user query, the IR agent uses the content of retrieved document file and two histories, Q/RDH and Q/SAH. Figure 5 depicts an example how the target agents are found with both histories, where Ⓐ to Ⓔ represent an IR agent, respectively. For simplicity, we assume here that the IR agent deals with the job of a HM agent. Furthermore, to make correspondence with a query and a retrieved document, we put a content file in Q/RDH.

Ⓐ has two query entries in its Q/RDH. Both queries were sent by Ⓐ itself. This figure shows that Ⓐ sent query 'Note PC' to Ⓒ and got the results from Ⓒ. Ⓒ recorded the query and Ⓐ's address into its Q/SAH. Ⓒ's address was recorded in a 'from' field of Ⓐ's Q/RDH. Since the content included in the results was the original of Ⓒ's user, Ⓒ's address could be seen in the 'original' filed of the content. In the same way, Ⓐ also sent query 'Wireless LAN' to Ⓓ, Ⓓ returned retrieved documents to it, and Ⓓ's address was recorded into the 'from' field of the same record as query 'Wireless LAN' in Ⓐ's Q/RDH. Since the documents included a content originally created by Ⓔ's user, Ⓔ's address could be seen in the 'original' field of the content.

If Ⓐ sends another query which is similar to 'Wireless LAN', say 'LAN', Ⓐ not only can find Ⓓ in a 'from' field of Q/RDH, but also find Ⓔ from an 'original' field of the content file by calculating a similarity between the query and the content file. Accordingly Ⓐ sends the query to both Ⓓ and Ⓔ.

The figure also shows that Ⓒ received query 'Mini Note PC' from Ⓑ, and both the query and Ⓑ's address were recorded into the Q/SAH. Although Ⓒ has not sent a query, it can find information related to the queries it received using its Q/SAH. Therefore when Ⓒ sends a query, say 'Note PC', it will find Ⓐ and Ⓑ with the Q/SAH and can send the query to both Ⓐ and Ⓑ, consequently.

2.4 The Effect of both Histories

As mentioned in the previous section, both Q/RDH and Q/SAH help to find target agents to send a query. If an IR agent can find enough number of the agents, no 'query multicasting' is carried out. Both histories, consequently, help to reduce communication loads between agents.

User's positive or negative judgments to the retrieved results could be embedded into them in Q/RDH, and these user evaluations are expected to be useful for finding target agents, which will return relevant information, with emerging a collaborative filtering effect. As an user creates more information, his/her IR agent can reply the retrieved results to more queries. Such IR agent therefore receives more queries from other agents. Thus, the agent achieves more information sources comprising of pairs of query and sender agent's address in its Q/SAH.

As the results, although the agent which has rich information receives more queries to be replied, it can achieve more information sources. That causes to emerge a 'give and take' effect.

3 Experiments

3.1 Implementation with KODAMA

The ACP2P method was implemented with Multi-Agent Kodama (Kyushu university Open & Distributed Autonomous Multi-Agent) [22]. Kodama comprises the hierarchical structured agent communities based on a portal-agent model. A portal agent(PA) is the representative of a community and allows all agents in the community to be treated as one normal agent outside the community. A PA has its role limited in a community, and the PA itself may be managed by another high-level portal agent. A PA manages all member agents in a community and can multicast a message to them. Any member agent in a community can ask the PA to do multicasting its message. All agents form a logical world which is completely separated from the physical world consisting of agent host machines. That means agents are not network-aware, but are organized and located by their places in the logical world. This model is realized with the agent middle-ware called Agent Communication Zone(ACZ for short). ACZ is primarily designed to act as a bridge between distributed physical networks, creating

an agent-friendly communication infrastructure on which agents can be organized in a hierarchical fashion more easily and freely. A Kodama agent consists of a kernel unit and an application unit. The kernel unit comprises the common basic modules shared by all Kodama agents, such as the community contactor or message interpreter. The application unit comprises a set of plug-in modules, each of which is used for describing and realizing a specialized or connatural function of agents.

3.2 Preliminaries

We used the Web pages of Yahoo! JAPAN[20] for the experiments. The Web pages used are broadly divided into five categories: animals, sports, computers, medicinal, and finance. Each of them consists of 20 small categories, which were selected in descending order of the number of Web pages recorded in a category. An agent is assigned to each category, and thus 100 IR agents are created and activated in the experiments. A category name is used as the name of an agent, and the Web pages in the category is used as the original documents of the agent as described in section 2.2. All agents are realized by describing their functions into plug-in modules of Kodama 's application unit[22]. From every category assigned to each IR agent, specific number (5 or 10) of common nouns are extracted in descending order of their occurrence frequency in the category except for the nouns consisting of one Kanji character. In order to distinguish the nouns in the Web pages, we used Japanese Morphological analyzer "ChaSen"[12]. All IR agents were assigned to the same one community for the simplicity.

We conducted experiments to show how both histories help to reduce communication loads between agents to look for relevant information to a query, and how Q/SAH helps to search for new information sources. For performing the experiments, we compared three methods : 1) ACP2P with a Q/SAH(wQ/SAH for short), 2)ACP2P without a Q/SAH(woQ/SAH for short), and 3) Simple method always employing a 'multicast' technique (MulCST for short).

3.3 Similarity Measure for Relevant Information to a Query

In order to find a request number of target agents to be sent a query, we calculated $Score(query, t\_agent)$, which returned the similarity value between query $query$ and a target agent t_agent, with equation (1); $Score(query, t\_agent)$ becomes higher if t_agent sends more number of similar queries and returns more documents related to $query$. After calculating $Score(query, t\_agent)$ for each IR agent t_agent in the Content file and both histories : Q/RDH and Q/SAH, the request number (RN) of target agents will be selected in the descending order of $Score(query, t\_agent)$, which value should be more than 0. Whenever RN of agents are not found, 'query multi-casting' technique will be employed with a portal agent. At that time, all answers will be replied to the portal agent as mentioned in section.

If a target IR agent finds relevant information to $query$, it returns a 'YES' message, otherwise a "NO" message. The judgment whether or not a document

is relevant to a query is decided according to the criterion of Boolean AND matching. That is, if the document includes all terms in *query*, it will be judged as relevant, otherwise irrelevant.

$$Score(query, t\_agent) = \sum_{i=1}^{k} \cos(\boldsymbol{query}, \boldsymbol{qh_{di}}) + \sum_{i=1}^{m} (\cos(\boldsymbol{query}, \boldsymbol{qh_{sai}}) + \varphi(i)) + \sum_{i=1}^{n} Sim_d(\boldsymbol{query}, \boldsymbol{doc_i}) \quad (1)$$

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

In equation (1), *query* consists of $w_1, ..., w_m$, and w_i $(1 \leq i \leq m)$ is a term in *query*. qh_d and qh_{sa} represent a query in a record of Q/RDH and one of 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_{sai}, which is *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_{sai} is directly sent by *t_agent*, δ is added to the score. In the experiment, we set it to 0.1 from our empirical experience. The last term $\sum_{i=1}^{n} Sim_d(\boldsymbol{query}, \boldsymbol{doc_i})$ is the total score of similarities between *query* and each of n documents originally created or owned by the user of *t_agent*. $Sim_d(query, doc)$ represents the similarity between *query* and the content of retrieved document *doc*. It is calculated with the following equation.

$$Sim_d(query, doc) = \sum_{i=1}^{m} \frac{tf_i}{tf_i + 1}$$

Where tf_i represents the occurrence frequency of w_i in *doc*.

3.4 Experimental Results

First, we conducted the experiment to show how much ACP2P with Q/SAH worked for reducing communication loads. To do that, we investigated the change of the average number of messages exchanged by each IR agent for every query sent by it. Each experiment was done with 10 queries for every IR agent, for 4 different request numbers : RN=3, 5, 7 and 10, and 2 different query length: QL=1 and 2. In each case, the average number of messages exchanged by each IR agent was being reduced for every query sent although we could not show the result due to the limitation of the space.

Next, for both QL=1 and 2, we compared the three methods: wQ/SAH, woQ/SAH and MulCST. RN was set to 10. The results are shown in figure 6.

For both cases, wQ/SAH had the best performance to reduce communication loads. It means that Q/SAH worked well to look up the relevant information with less communication efforts, and caused the positive feedback effect.

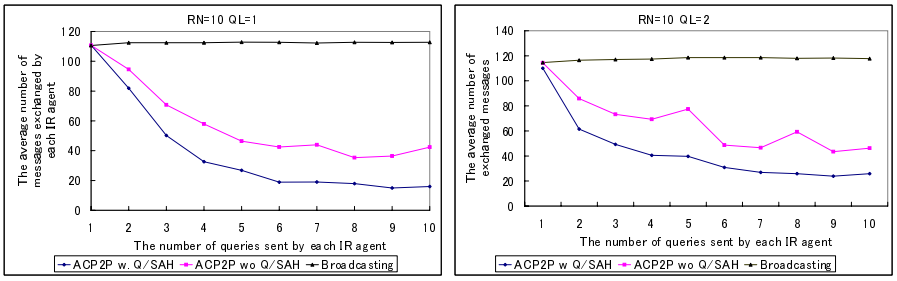

Fig. 6. The average number of messages exchanged by each IR agent for each query sent, where QL=1 in the above, and QL=2 in below. RN=10 in both cases.

We also compared three methods for the average number of acquired documents of each IR agent. The results were shown in table 4. Except for the case of RN=3 of QL=2, the difference between wQ/SAH and MulCST was little.

Table 4. Comparison on average number of acquired documents when query length is 1 (left) and 2 (right)

QL=1, RN=	3	5	7	10
wQ/SAH	269.1	385.3	443.0	497.6
woQ/SAH	258.8	331.6	424.9	476.4
MulCST	233.8	366.4	421.3	487.0

QL=2, RN=	3	5	7	10
wQ/SAH	54.9	126.3	178.9	226.8
woQ/SAH	54.8	96.8	150.0	208.3
MulCST	85.3	148.4	191.0	232.6

4 Related Work

There are lots of related work to the topics of this paper, such as distributed information retrieval(DIR), P2P file searching, collaborative filtering and so forth. The DIR selects some of IR systems to send a query, aggregates the results replied by the selected IR systems, and presents them to a user. Before selecting the IR systems to be sent a query, the resource description of each IR system is often created using such as query-sampling based method[3]. For ACP2P method, history Q/RDH is expected to cause the same effect of the resource description, and furthermore, history Q/SAH works as good heuristics to find relevant information.

A lot of P2P file searching systems such as Freenet[4], Chord[19], Gnutella[5] and Napster[14] have been proposed. Freenet and Chord are carried out in a pure P2P computing architecture. They neither employ 'broadcast' techniques like Gnutella, nor have a centralized server machine like Napster. Freenet provides information sharing and information finding functions among anonymously distributed nodes. Although Chord does not provide anonymousness of nodes,

it has an efficient protocol of looking up nodes. Their node searching strategies are conducted according to keywords attached in the information of the nodes. On the contrary, ACP2P method makes use of the content information of documents, and two histories: Q/RDH and Q/SAH to search for target agents with relevant information. In particular, Q/SAH provides similar effects to the link analysis like PageRank[2] or HITs algorithm[10] and makes a natural collaborative filtering effect emerge.

Although lots of work on the field of Collaborative Filtering (e.g.[6],[15],[18],[1],[8],[13],[16]) have been performed, most of them assume the server-client computational model and need the procedure to collect all data from other nodes explicitly. The ACP2P method takes a distributed data management method with agent communities based on a P2P computing architecture, and makes a natural collaborative filtering effect emerge, with both Q/RDH and Q/SAH.

5 Conclusion and Future Work

We discussed an agent community based information retrieval method, called ACP2P method, which used the content of retrieved document file and two histories: Q/RDH and Q/SAH to find target agents to be sent a query. The method was implemented with Multi-Agent System Kodama.

We conducted several experiments to show whether or not both histories helped to reduce communication loads between agents to search for relevant information to a query, and whether or not Q/SAH helped to look up new information sources. The experimental results showed the efficiency of ACP2P method and the usefulness of both histories for looking up new information source. We also investigated and confirmed that the number of agents exchanging query messages together was increased by the effect of history Q/SAH although we could not describe the detail about it due to the limitation of the space.

We are currently preparing to investigate the accuracy or any ranking method of retrieved results, especially, how we make use of the other user's feedbacks embedded into the results. Developing an effective method for creating hierarchical agent communities to allocate agents into at the initial stage, is future work.

Acknowledgment

This research was partly supported by the Telecommunication Advancement Organization(TAO) of Japan, under the grant to "the Research on Management of Security Policies in Mutual Connection".

References

1. M. Balabanovic and Y. Shoham. Content-based, collaborative recommendation. *Communications of the ACM*, 40(3), 1997.

2. S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proc. of 7th International World Wide Web Conference:WWW7 Conference*, 1998.
3. J. Callan and M. Connell. Query-based sampling of text databases. *ACM Transactions on Information Systems*, 19(2):97–130, 2001.
4. I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. 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.
5. Gnutella. http://gnutella.wego.com/, 2000.
6. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35:61–70, 1992.
7. N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. M. Sarwar, J. L. Herlocker, and J. Riedl. Combining collaborative filtering with personal agents for better recommendations. In *AAAI/IAAI*, pages 439–446, 1999.
8. J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR99*, pages 230–237, 1999.
9. A. Kanfer, J. Sweet, and A. Schlosser. Humanizing the net: Social navigation with a "know-who" email agent. In *The 3rd Conference on Human Factors & The Web, http://www.ncsa.uiuc.edu/edu/trg*, 1997.
10. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46:604–632, 1999.
11. K. Lang. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.
12. Y. Matsumoto, K. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara. Japanese morphological analyzer "chasen" ver. 2.3.1, http://chasen.org/, 2003.
13. P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In *SIGIR-2001 Workshop on Recommender Systems*, 2001.
14. Napster. http://www.napster.com/, 2000.
15. P.Resnick, N.Iacovou, M.Suchak, P.Bergstrom, and J.Riedl. Grouplens: Open architecture for collaborative filtering of netnews, 1994.
16. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW10*, pages 285–295, 2001.
17. J. B. Schafer, J. A. Konstan, and J. Riedi. Recommender systems in e-commerce. In *ACM Conference on Electronic Commerce*, pages 158–166, 1999.
18. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 210–217, 1995.
19. I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. 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*, pages 149–160, 2001.
20. Yahoo. http://www.yahoo.co.jp/, 2003.
21. D. Yimam-Seid and A. Kobsa. Expert finding systems for organizations: Problem and domain analysis and the demoir approach. *Journal of Organizational Computing and Electronic Commerce*, 13(1):1–24, 2003.
22. G. Zhong, S. Amamiya, K. Takahashi, T. Mine, and M. Amamiya. The design and application of kodama system. *IEICE Transactions INF.& SYST.*, E85-D(04):637–646, 4 2002.