IEICE TRANS. COMMUN., VOL.E89–B, NO.9 SEPTEMBER 2006

PAPER Special Section on Networking Technologies for Overlay Networks

A Dynamic Index Allocation Scheme for Data Retrieval and Provision in Peer-to-Peer Networks

Atsushi ITO[†a)], **Tomoyuki OHTA**[††], ***Members***, **Kouichi MITSUKAWA**[††], ***Nonmember***, ***and*** **Yoshiaki KAKUDA**[††], ***Member***

SUMMARY File-sharing Peer-to-Peer systems are effective for autonomous data retrieval and provision over the networks. However, the early data retrieval schemes such as Gnutella and Local Indices have low performance and large overhead. In order to solve weakness of early schemes, this paper proposes a dynamic scheme for data retrieval and provision, in which indices are adaptively allocated in appropriate nodes to variation of traffic patterns caused by query messages. The simulation experimental results show that the proposed scheme has good performance with reasonable overhead even when the traffic patterns vary as time proceeds.

key words: *peer-to-peer system, data retrieval, data provision, index allocation*

1. Introduction

Recently, Peer-to-Peer (P2P) systems become popular along with increasing computational power of nodes and transmission speed of communication links [17]. Napster [3], Gnutella [2], and Freenet [1] have been developed as early P2P systems. Napster is a hybrid P2P structure since it uses client-server structure to retrieve data, while Gnutella and Freenet are pure P2P structure. Moreover, other P2P systems such as Chord [9], CAN [8] and Pastry [5] were proposed. These P2P systems use distributed hash table to retrieve data contained in each node. Gnutella, which is one of the representative file-sharing P2P systems, makes logical connections among nodes, and constitutes a mesh type logical network without any central node that manages the whole network.

Therefore, Gnutella has properties as easy installation and provide a feature of fault tolerance. The search schemes used in Gnutella are, however, inefficient, since Gnutella floods each query over the whole network. So that, the response is slow. In order to solve these defects, Local Indices scheme was proposed [10]. In this scheme, each node maintains an index over the data of all nodes within some hops from itself. Each node can respond to a Query message in shorter time referring the index and suppress unnecessary query. Local Indices scheme can prevent wasting network resources. Not all indices are, however, used effectively since Query messages to retrieve data are not uniformly generated in network. There are high possibilities that Local Indices scheme wastes resources to create and maintain indices that are not used much frequently.

In order to reduce consumption of resource and retrieval time, we have proposed the dynamic node allocation scheme to provide efficient data index mechanism for P2P data-sharing system [6], [15]. In P2P systems, a node always joins in or moves out of the network. So that, the indices to data have to be updated according to the change of nodes that have data. If there are a lot of incorrect indices in the network, users cannot obtain data based on the indices and a lot of Query messages might be generated to retrieve data. Accurate indices to data are very important in the distributed systems such as P2P systems. Therefore, we propose a new "dynamic index allocation scheme" to provide the reliable indices to data for P2P data sharing system.

In this paper, the effectiveness of the proposed scheme is described comparing with Gnutella and Local indices schemes through the simulation experiments. In Sect. 2, we explain related works and our intention. In Sect. 3, the proposed scheme is described in detail. In Sects. 4, 5 and 6, effectiveness of the proposed scheme is explained based on simulation results.

2. Related Works and Our Intention

2.1 Gnutella

In Gnutella, a node discovers other nodes in which required data are stored then retrieves that data. In order to discover such nodes, Ping and Pong message are used. A node broadcasts a Ping message to discover nodes in which required data are stored. The Ping message is forwarded over the network within Time To Live (TTL). When a node receives a Ping message, the node responds by sending a Pong message. A Query and QueryHit message are used to retrieve the data. A node, which wants to retrieve the data, broadcasts a Query message to nodes within the TTL. When a node that has the data receives the Query message, the node sends a QueryHit message back to respond to the Query message.

2.2 Local Indices

In Local Indices scheme, each node maintains an index over

Manuscript received December 19, 2005.
Manuscript revised April 5, 2006.

[†]The author is with the KDDI Corporation, Tokyo, 102-8460 Japan.
[††]The authors are with the Faculty of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.
a) E-mail: at-itou@kddi.com

DOI: 10.1093/ietcom/e89–b.9.2336

the data of all nodes within r hops of itself. A node that receives a Query message can respond by using the index in which information of the location of data is collected from the neighboring nodes and stored as metadata. Therefore, the search time of Local Indices scheme is shorter than that of Gnutella.

Creating Indices: When a node connects to a P2P network, it broadcasts a Join message to nodes within r hops of itself. The Join message contains metadata that has information of data stored in the node. A node that received the Join message registers the metadata in the Join message to the index and sends back the metadata that the node has. Each entry in the index consists of the metadata.

Maintaining Indices: There are two cases to update an index. The first case is that a node updates data by addition, deletion and modification. In this case, the node generates an Update message that contains the updated information and broadcasts it to nodes within r hops of itself. When a node receives the Update message, the index is updated. The second case is that a node leaves the P2P network. Each node always exchanges Ping and Pong messages with the neighboring nodes to confirm the connectivity. If a node cannot receive a Pong message in a specified period after sending a Ping message to another node, the node deletes the metadata of that node.

2.3 Techniques Using Indices and Our Intention

There are several techniques proposing efficient algorithm using indices. We will briefly explain four typical techniques and address what we want to propose in this paper. [11] proposes a technique to create groups of semantically related nodes based on category of information in a node. [12] proposes a technique called Routing Indices to forward queries to neighbors that are more likely to have answer. The Indices are stored in each node. [13] uses supernodes technology proposed by KaZaA [16]. [13] adds topology adaptation algorithm to add neighbors in a node. Each node has capacity. Capacity controls the number of neighbors to be registered. [14] introduces interest based shortcut which is an overlay link to a particular contents.

These approaches propose several kind of "indices," however, mechanism to select a node that has indices are not well formalized. For example, in [14], shortcut is selected randomly and in [12], routing indices are stored in each node. We thought that selecting a node that has indices is important to improve the performance. So that, we propose the index node selection scheme based on Query and QueryHit messages as described in Sect. 1.

3. Proposed Scheme

In the Local Indices scheme, all nodes exchange metadata with the neighboring nodes and create indices to data. The metadata includes data ID and location. This scheme may waste network resource to exchange too much metadata. Indexes to many popular files are stored in each node, so that index information stored in each node is overlapped. This will produce heavily load for the index peer and may reduce performance of the overlay network. Also, some data are accessed very often but other data are accessed very rare, however metadata exchange cost is equivalent. Therefore, we propose a new dynamic index allocation scheme for P2P data sharing system. In the proposed scheme, according to the occurrence of Query and QueryHit messages, each node determines dynamically whether it should create indices or not. Consequently, it can avoid creating useless indices and achieve to use bandwidth effectively.

3.1 Dynamic Index Allocation

This section describes the mechanism to determine dynamically whether each node should create indices. Hereinafter, we call a node that has indices "index node" and the other node "normal node," respectively.

- Initially, all nodes in the network do not have any indices.
- Each node has two thresholds, UPPER and LOWER (UPPER > LOWER).
  The mechanism to decide UPPER and LOWER is explained in Sect. 3.6.

Each node measures the number of Query and QueryHit messages that has received for a data interval.

Based on the measurement, each node decides whether it has the index to data or not. Indices to data should be allocated in nodes that have received many Query and QueryHit messages. To decide index node based on Query and QueryHit messages, the following procedures (1), (2), and (3) are periodically performed in each node. They make possible to allocate indices dynamically to change of occurrences of Query messages.

(1) Calculate proper threshold value
Based on the number of Query message and QueryHit message measured in each node, the node calculates proper threshold value. We introduce two methods to calculate proper threshold value by using the following equations.

P-(a) $Query + (n-1) \times QueryHit$

P-(b) $n \times QueryHit \div Query \times 100$

(n: the number of neighboring nodes)

Now, we call result of P-(a) and P-(b) as "proper value." Here, if the number of Query messages that a node has received is zero, result of P-(a) and P-(b) of the node are zero. Hereinafter, we call result of P-(a) and P-(b) as P-(a) and P-(b) respectively if the context allows.

In P-(a), a node which frequently forwards either Query or QueryHit message recognizes that there are nodes which frequently request data in the vicinity of the node, and becomes index node. On the contrary, in P-(b), a node such that the ratio of the number of Query messages to the number of QueryHit messages is high becomes index node.

(2) Compare proper value with the threshold UPPER

If the proper value is larger than threshold UPPER in a node;

- A normal node is changed to index node and creates indices, and
- An index node keeps current status.

(3) Compare proper value with the threshold LOWER
If the proper value is smaller than threshold LOWER in a node;

- An index node is changed to normal node and discards indices, and
- A normal node keeps current status.

3.2 Creating Indices

The proposed scheme has two mechanisms to register the metadata in the indices.

3.2.1 Base Mechanism for Creating Indices

The base mechanism is that an index node obtains metadata from all nodes within r hops of itself to create an index like Local Indices scheme. The index node generates an Index-Query message to obtain metadata and broadcasts it to all nodes within r hops of itself. Each node that received the Index-Query message contains metadata stored in the node to an Index-Reply message and returns it to the index node. The index node registers the metadata that are contained in the Index-Reply into the index in it. Here, we call a set of an index node and all nodes within r hops of itself as "information group." The index node manages its information group.

3.2.2 Additional Mechanism for Creating Effective Indices

In the base mechanism, each index node can obtain metadata only from the nodes within r hops of itself. Therefore, it is expected that reduction of the number of Query messages and the search time only using the base mechanism is not so effective because the distance between the index node which has an index to data and the node which has the data is rather short.

In order to solve the above defect, the additional mechanism is introduced for the proposed scheme. We assume that the index node can obtain the metadata in the index pointing to data stored in nodes that are far from more than r hops of itself. In the additional mechanism, each node that received a QueryHit message checks the source node of the QueryHit message. If the QueryHit message is sent from the outside of the information group to which the node belongs, the node derives the metadata from the QueryHit message, generates a Report message, and sends it to index nodes in the information group to which the node belongs. The Report message is only generated for a QueryHit message that is passing on an information group, not generated for a QueryHit message whose source node is in the same information group.

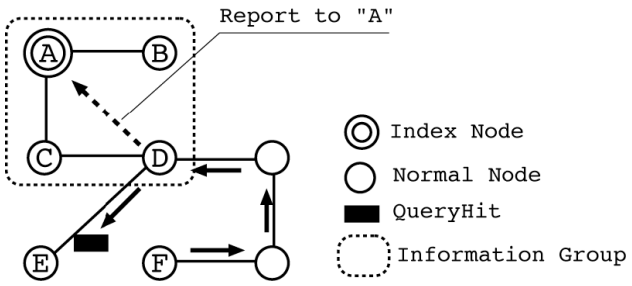

Fig. 1 Registration of metadata contained in QueryHit.

We explain the base and additional mechanisms in the proposed scheme using an example. As shown in Fig. 1, the index node A creates an index based on metadata collected from nodes B, C, and D within 2 hops of itself and the information group consists of A, B, C, and D by the base mechanism. Suppose that node E broadcasts a Query message to retrieve data and the data are discovered at node F. In this case, node Freturns a QueryHit message to node E. When node D receives the QueryHit message, it checks the source node of the QueryHit message and knows that the Query message is sent from the outside of the information group to which node D belongs. Therefore, node D derives the metadata from the QueryHit message, generates a Report message, and sends it to node A. Node A that received the Report message registers the metadata into the index. By using these two mechanisms, the proposed scheme can create the indices that have metadata collected from the wide range of nodes in comparison with Local Indices scheme. It is expected that the proposed scheme can considerably reduce the search time and the number of Query messages. The idea of index node and information group is different from the idea of super node of KaZaA [16]. A super node has pointer to data stored in peers associated with super node. On the other hand, as explained above, the index node has not only indexes of information stored in normal node but also has indexes of information stored in nodes outside the information group.

3.3 Maintaining Indices

Five messages, Update, Report, Logout, Release and missHit, are used to maintain indices. A node that belongs to an information group sends an Update message to the index node when the data in that node are updated, deleted or added. The index node that received the Update message updates the indices based on the information contained in the Update message. In addition, when a node that belongs to an information group moves out of the network, the node sends a Logout message to the index node. The index node deletes the indices for the data of the node. We introduced Report message to make index mechanism work efficiently. As described in Sect. 3.2, several QueryHit messages run through an information group. If an information group can

Table 1 Messages.

	Dynamic index allocation scheme	Local Indices scheme
Basic function	Query QueryHit Index-Query Index-Reply	Query QueryHit Index-Query Index-Reply
Introduced in our scheme	Report Update Logout Release missHit	

gather QueryHit information outside the information group and send Report to notice this inforamtion, the index node can get more information of location of files. Since the proposed scheme introduces a Report message, the metadata on the outside of the information group can be registered in the indices of the index node. The proposed scheme needs to update indices that are created based on the metadata obtained from the outside of the information group when the metadata are updated or deleted. When a node tries to get data for such metadata based on the QueryHit message sent from an index node, the node can not get the data because the data had already deleted or the node which has the data had moved out of the network. In this situation, the node that could not get the data sends a missHit message to the index node. The index node that received the missHit message deletes the entry from the indices to update the indices. In the Table 1, we summarize difference between our dynamic index allocation scheme and Local Indices scheme. Our proposed scheme has features of getting and maintaining indices that belong to out of an information group efficiently by introducing five messages such as Report message, Update message, Logout message, Release message and missHit message.

3.4 Dependability Feature

In a P2P network, each node is not stable, since some may be disappear when they are turned off or when communication link failure happens. In these cases, the network has to keep dependability to provide stable services. Report message, Update message, Logout message and missHit message have function not only for maintaining indices but also keeping dependability of a P2P network. For that purpose, the maintaining indices function, described in Sect. 3.3, can work effectively. For example, when a node is shut down, a Logout message is sent to the index node. So that, other node can understand that node has been out of the network and it is prevented to sending Query message to that node. Also, if a link failure happened somewhere in a network, message transfer route might be changed and new QueryHit message may be observed in a node. Then a Report message may generate and an index node add a new entity in indices. The network can suppress possibility of missHit.

In the proposed scheme, we assume that Report message, Update message, Logout message and missHit message are not lost in the network. It is possible to find lost of Logout message, for example, by defining maximum number of missHit relating an index node. Detailed discussion in problems of failures, however, is further study.

3.5 Release of the Information Group

When the proper value that each node calculates for a time interval is less than the threshold LOWER, an index node is changed to a normal node. At this time, the index node needs to release the information group that it has held until then. Also, the index node sends a Release message to all nodes in the information group. The node that received the Release message from the index node leaves from the information group and stops to send a Report message to the index node.

In addition, an index node needs to release the information group when the index node moves out from the network. At this time, the index node sends a Logout message to all nodes in the information group. The node that received the Logout message leaves from the information group and stops to send a Report message to the index node.

For example, when node *C* in Fig. 1 leaves from the information group, a Logout message is sent to the index node *A*.

3.6 Adaptive Threshold

In the proposed scheme, index nodes are allocated based on the proper value, which is calculated using the numbers of Query and QueryHit messages, and the UPPER and LOWER threshold. However, the traffic is not uniformly caused by Query and QueryHit messages. The adaptive threshold mechanism should be introduced to adaptively set the thresholds according to the traffic situation in the vicinity of a node. The thresholds are adaptively changed as follows. Each node periodically calculates the proper value and records it. After each period of time, the node calculates the average and deviation numbers. If the average number is zero, it is judged that the node had not received any messages from the neighboring nodes, and thus the thresholds are not changed. Then, the thresholds UPPER and LOWER are set to the value that is the average number plus the deviation number and the value that is the average number minus the deviation number, respectively. When the new UPPER and LOWER are set, the node discards the recorded proper value. If the new threshold LOWER becomes less than zero, the node sets the new threshold LOWER to one in case of P-(a) and the node does not change the thresholds LOWER and UPPER in case of P-(b).

4. Simulation Experiment 1

In order to show the effectiveness of the proposed scheme in comparison with Gnutella (version 0.6) and Local Indices

Table 2 Simulation model.

Number of nodes	1000
Number of neighboring nodes	4 - 5
Delay time between links	10ms
Frequency of Sending Query messages	10 nodes/sec.
Time-to-live of Query message	7
Sending interval of Ping message	30 sec.
Simulation time	500 sec.

Table 3 The number of data in a node.

Type	The number of data	Ratio
Type1	0	25%
Type2	5	20%
Type3	50	30%
Type4	100	25%

scheme, the simulation experiments are conducted. The network simulator 2 (version 2.26) [4] is used for the simulation experiments.

4.1 Simulation Model

As time proceeds, a node moves out of the network and instead of it, a new node joins in the network. Thus, the number of nodes in the network is always not changed. In addition, a node that moved out of the network cannot join in the network again. This is because it assumes that the invalid entries in the index node do not increase. In the simulation experiments, one node moves out of the every one second to 30 seconds. Initially, any copy of data network and one new node joins in the network does not exist in the network. However, when each node finds data by Query and QueryHit messages, the node gets the copy of the data. The three kinds of the network topologies are generated based on Table 2. In the simulation experiments, nodes totally retrieve the data 5,000 times. The number of data in a node is classified into four types as shown in Table 3. We applied the same way as described in [7] to define the number of data in a node. Type and size of date are not considered in this simulation. In order to show the effectiveness of the proposed scheme, we performed the simulation for the following schemes.

(1) Gnutella
(2) Local Indices
(3) P-(a), (initially LOWER:20, UPPER:50)
(4) P-(b), (initially LOWER:10, UPPER:30)

In Local Indices scheme, the radius of the index node to create the indices is $r = 1$. In our dynamic index allocation scheme, we introduced P-(a) and P-(b) to calculate the proper value, respectively. The proper value is calculated every 5 seconds and the thresholds UPPER and LOWER are set every 50 seconds. In addition, we assume that Report/Update/Logout/missHit messages are not lost as described in Sect. 3.4.

Table 4 Simulation results.

	Search time	Number of messages	Number of entries	Hit ratio
Gnutella	96ms	6,537,475	0	92.3%
LI	77ms	6,336,287	149,907	96.6%
P-(a)	58ms	5,736,170	134,287	95.7%
P-(b)	60ms	5,837,984	129,937	95.6%

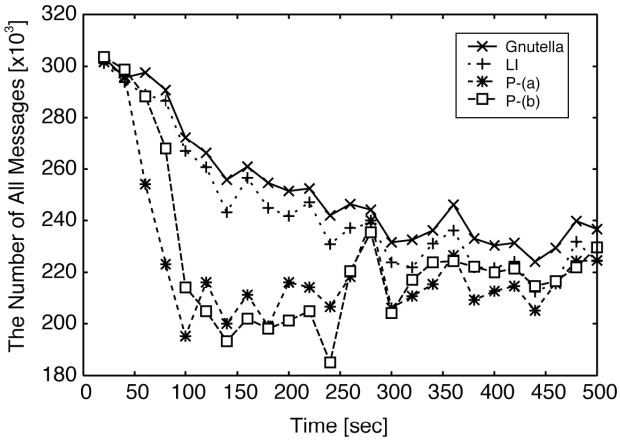

Fig. 2 The number of all messages (the *Y* axis starts from 180).

4.2 Simulation Results and Observations

Table 4 shows the simulation results. As shown in Table 4, the search times of the proposed schemes, P-(a) and P-(b), are shorter than that of Local Indices scheme. In addition, the proposed schemes could reduce at most 10% of the number of entries in comparison with Gnutella and Local Indices schemes. The hit ratios of Local Indices scheme and the proposed scheme are almost the same. Consequently, we can say that the simulation results show the effectiveness of the proposed schemes.

4.2.1 The Number of Messages

Figure 2 shows the number of all messages in every 20 seconds. As shown in Fig. 2, the proposed schemes, P-(a) and P-(b), reduce the total number of messages at the early period of the simulation in comparison with Gnutella and Local Indices schemes. Figure 3 shows only the number of Query messages. As shown in Fig. 3, the numbers of Query messages of the proposed schemes are always much less than those of Gnutella and Local Indices scheme. As shown in Fig. 2, however, after around 250 seconds, the numbers of the proposed schemes and Local Indices become closer. In the proposed scheme, Report messages are heavily generated and each node in information group sends the metadata contained in QueryHit messages to the index node. So

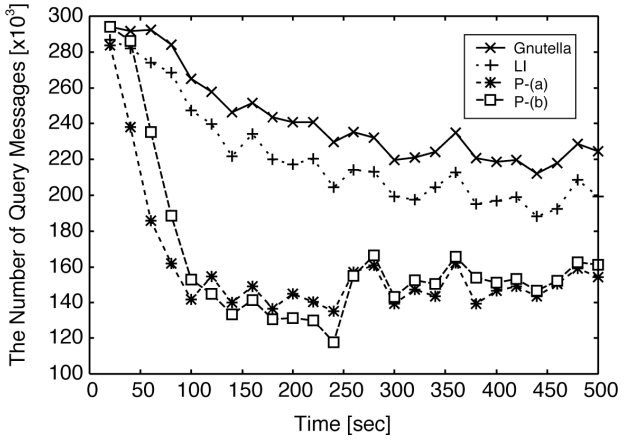

Fig. 3 The number of Query messages (the *Y* axis starts from 100).

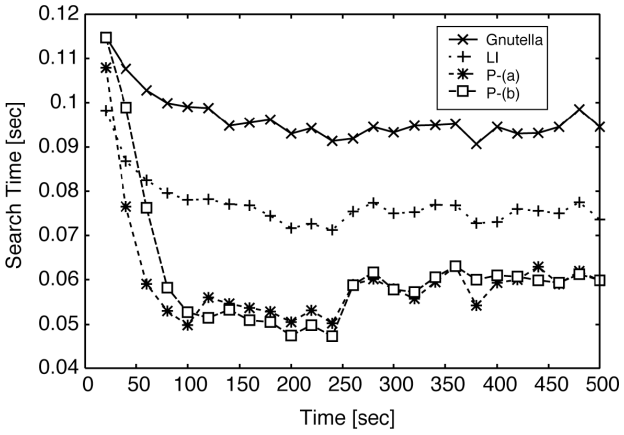

Fig. 4 Search time (the *Y* axis starts from 0.04).

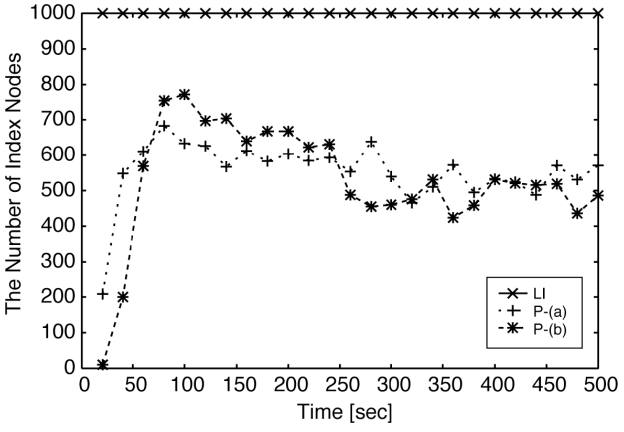

Fig. 5 The number of index nodes.

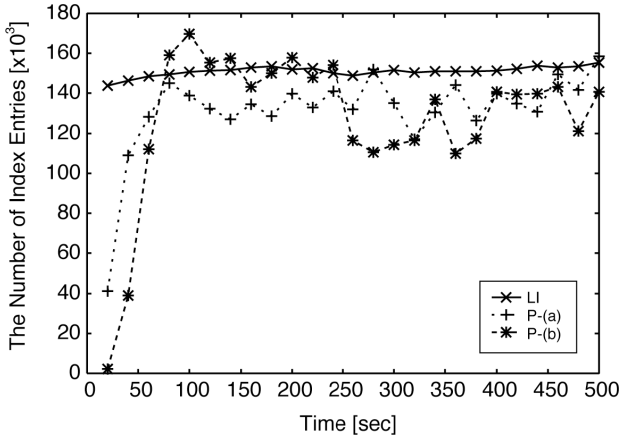

Fig. 6 The number of entries in index nodes.

that, according to increase of the numbers of index nodes in the network or QueryHit messages, Report messages are generated more frequently. The proposed schemes could reduce only at most 10% of the number of messages, however, the proposed schemes can reduce the number of Query messages at all times in comparison with Gnutella and Local Indices schemes.

It is expected that index nodes are adequately allocated over the network. Each index node can get the metadata from nodes that are far from the index node by Report messages. Therefore, since each index node can create and maintain many indices by Report messages, there are high possibilities that each node can obtain the indices to data without forwarding Query messages from the index node. On the other hand, in case of the number of all messages, the proposed schemes can reduce the number of all messages at the early period of the simulation. However, after the middle period of the simulation, the numbers of all messages of the proposed schemes are almost the same as those of the conventional schemes. In Gnutella and Local Indices schemes, the number of Query messages decreases because of the copy of data and the change of the network topology according to progress of simulation, while in the proposed schemes the numbers of Report and missHit messages which are continuously utilized to update the indices increase.

4.2.2 Search Time

Figure 4 shows the search time every 20 seconds. At the beginning of the simulation, the search time of the proposed scheme is the same as that of Gnutella. This is because in the proposed schemes there are initially no index nodes in the network. After around 60 seconds, the search time of the proposed schemes is always shortest among the all schemes.

4.2.3 The Number of Index Nodes and the Entries

Figure 5 shows the number of index nodes in the network. In case of Local Indices scheme, all nodes that join in the network behave as index nodes. The number of index nodes is thus always 1,000. In case of the proposed schemes, the number of index nodes is between 400 and 800. Figure 6 shows the number of the total entries that index nodes hold. In case of Local Indices scheme, all nodes always create and maintain indices by exchanging the metadata with the neighboring nodes. On the contrary, in case of the proposed

schemes, since index nodes create the indices according to the occurrence of Query messages, nodes of the numbers of entries of the proposed schemes are less than that of Local Indices scheme. The numbers of index f the proposed schemes are half as many as that of Local Indices scheme. In addition, the numbers of entries of the proposed schemes are also less than that of Local Indices scheme. Despite that in the proposed schemes the numbers of index nodes and entries which all index nodes maintain become smaller, efficient reduction of Query messages and search time are achieved. Using Report messages, each index node could create the indices based on the metadata that are sent from far nodes. Consequently, as the number of entries which each index node holds increases, the index node can give the indices to data stored in the far nodes to the vicinity of the index node.

4.2.4 The Number of Valid and Invalid Entries

In the simulation experiments, a node, that sent Query messages to retrieve data, accesses to the node that is indicated by the entries contained in the QueryHit message and obtains the data. When the node deletes or updates data stored in it or the node moves out of the network, however, the entries to the data obtained from the node by the QueryHit messages become invalid. Figure 7 shows the numbers of valid and invalid entries. When a node accesses to the node that is indicated by the entry contained in the QueryHit message and the node obtains the correct data, the entry is defined as the valid entry. On the other hand, if the node cannot obtain the correct data, the entry is defined as the invalid entry.

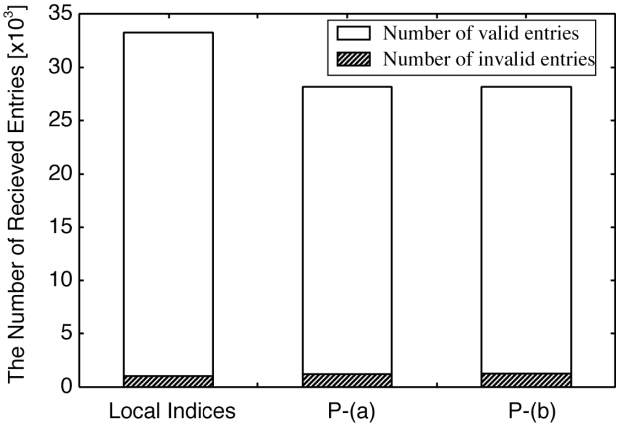

Fig. 7 The number of valid and invalid entries.

The simulation results show that the numbers of entries by the proposed schemes are less than that by Local Indices scheme. It is because in the proposed schemes, there are high possibilities that index nodes can find the data using indices of themselves. In fact, in Local Indices scheme, index nodes could find data using indices of themselves 129 times while in the proposed scheme index nodes could find data using indices of themselves 430 times. Such index nodes need not to send Query messages. In the proposed scheme, the number of Query messages generated by index nodes can be restrained. As a result, entries in the QueryHit message received by index nodes become decreased. As long as we see the numbers of valid entries and invalid entries in Fig. 7, the hit rate of successful data retrieval by the proposed schemes is almost the same as that by Local Indices scheme. Therefore, we can say that data are successfully retrieved and provided with a smaller number of entries in the proposed schemes.

4.2.5 Overhead

The proposed scheme has overhead because of five additional messages such as Report/Update/Logout/Release/missHit messages comparing to Local Indices method. Figure 2 shows that even these overhead, total numbers of messages of out scheme is lower than Gnutella and Local Indices. Figure 3 to Fig. 7 explain that the proposed method is enough efficient.

4.2.6 Volume of Messages

Table 5 shows that the total volume of messages is equivalent for Local Indices scheme and P-(a) and P-(b).

The average number of QueryHit is almost twice larger than that of Local Indices scheme. It is expected that more QueryHit reduce search time, so that our proposed schemes have better ability to retrieve data.

We calculated this result under the following condition. We consider Query, QueryHit, IndexReply, Report, Update that are relating to information retrieval. Ping/Pong and other messages that are used commonly in each scheme are not considered to make discussion simpler.

(1) Query, QueryHit, Report have only one index information.

(2) Each node has 50 indices. (In this experiment, total

Table 5 Volume of messages.

	The average number of Query	The average number of QueryHit	Data size (n)	The number of data/ node	[Index Reply]	[Report]	[Update]	Volume of messages
LI	5,760,594	462,943	1	50	4,951	0	16,393	6,487,497
P-(a)	4,077,530	885,380	1	50	20,330	563,543	9,961	6,552,931
P-(b)	4,232,826	847,130	1	50	21,925	543,766	9,518	6,729,507

50,000 data is assigned in 1,000 nodes.)

(3) Volume of message = the number of Query message $\times n1$ + the number of QueryHit message $\times n2$ + the number of IndexReply message $\times 50 \times n2$ + the number of Report message $\times n2$ + the number of Update message $\times n2$ ($n1$, $n2$: message size)

The size of retrieval keyword stored in a Query message ($n1$) and the size of filename stored in other messages such as QueryHit message ($n2$) are equal since retrieval keyword might be a file name. Then we can introduce the following equation.

(4) Volume of message = (the number of Query message + the number of QueryHit message + the number of IndexReply message $\times$ 50 + the number of Report message + the number of Update message) $\times n$

From (4), volume of message depends on the number of messages and the message size, however, not depends on schemes. So that, we define n=1 in this simulation. Some small amount of control messages is ignored in this discussion.

In the proposed schemes, the number of Query is decreased, however, the number of IndexReply and Report is increased. When index node is replaced, the new index node asked to neighbor node to send index information to the new index node. We can say that proposed schemes can get index beyond neighbors using Report.

We can conclude that our proposed schemes and Local Indices scheme require almost same amount of messages, however the retrieval time is reduced. So that, both proposed schemes are better than Local Indices scheme.

4.2.7 Effect of Report Message

Figure 7 explains the difference of entries. The number of entries of Local Indices scheme is 33,107, of P-(a) is 27,913 and of P-(b) is 28,168. The number of entries for P-(a) and P-(b) is less than the number of entries for Local Indices scheme, however, the number of invalid entries is 967 for Local Indices scheme, 1,338 for P-(a) and 1,341 for P-(b). So that, proposed schemes can generate indices efficiently.

As described in explanation of the number of valid and invalid entries, in Local Indices scheme, index nodes could find data using indices of themselves 129 times while in the proposed scheme index nodes could find data using indices of themselves 430 times. However, as shown in Fig. 7, the number of entries of Local Indices scheme is larger than that of proposes schemes. If proposed scheme does not have the Report function, the number of entries might be same. So that, we can explain the difference of the number of data retrieval only using indices in a index node explains the effectiveness of proposed schemes.

In addition, as shown in Table 4, search time of the proposed schemes is less than that of Local Indices scheme. This result means that using indices for information in other information group makes it possible to realize fast information retrieval.

5. Simulation Experiment 2

In simulation experiment 1, comparison between P-(a) and P-(b) was not explained. In order to show a difference between them, the simulation experiments with respect to the occurrence of Query messages have been conducted.

5.1 Simulation Models

As shown in Tables 1 and 2, the simulation models are the same as those of simulation experiment 1 except for change of frequency of sending Query messages. As simulation proceeds, the frequency of sending Query messages is changed as shown in Table 6. Nodes that send Query messages are randomly selected.

The initial thresholds LOWER and UPPER in P-(a) are (LOWER:20, UPPER:50) and in P-(b) are (LOWER:10, UPPER:30). The radius of the index node to create the indices is r = 1, the proper value is calculated every 5 seconds and the thresholds UPPER and LOWER are set every 50 seconds, which are the same as in simulation experiment 1.

5.2 Simulation Result

5.2.1 The Number of All Messages

Figure 8 shows the number of all messages every 20 seconds. Along with the increase of the occurrence of Query

Table 6 The frequency of sending Query message in a second.

Interval	The number of nodes that send Query Message in a second
0 – 100 seconds	2
100 - 200 seconds	5
200 - 300 seconds	10
300 - 400 seconds	5
400 - 500 seconds	2

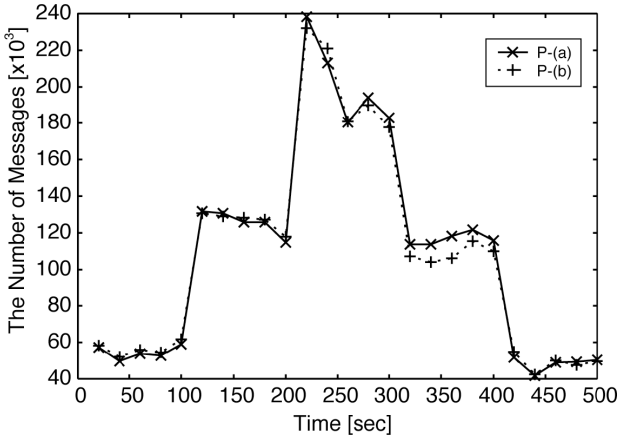

Fig. 8 The number of all messages (the Y axis starts from 40).

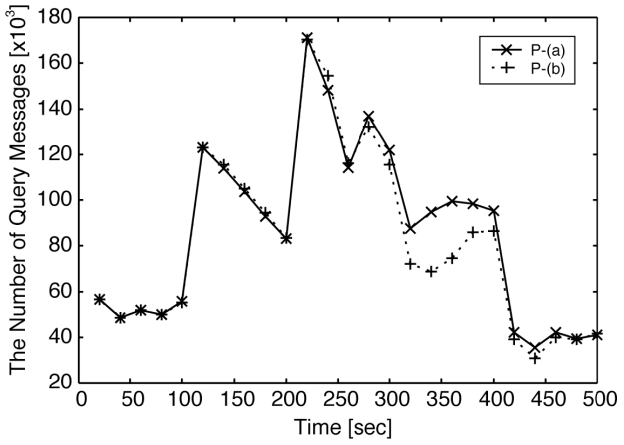

Fig. 9 The number of Query messages (the *Y* axis starts from 20).

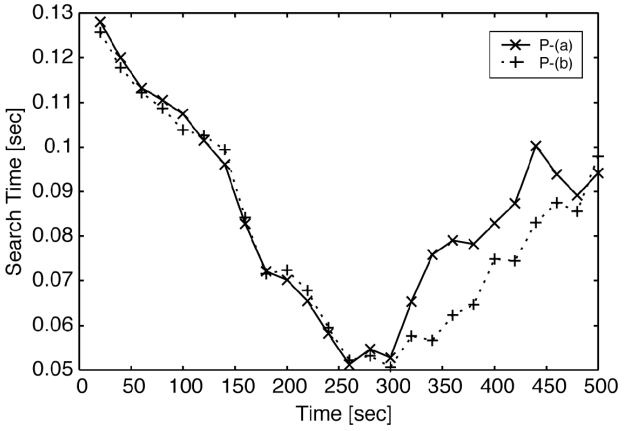

Fig. 10 Search time (the *Y* axis starts from 0.05).

messages, the numbers of all messages of P-(a) and P-(b) increase. The difference between P-(a) and P-(b) is not clear with respect to the number of all messages.

Figure 9 shows only the number of Query messages. During 300–400 seconds, the number of Query messages of P-(b) is less than that of P-(a).

5.2.2 Search Time

Figure 10 shows the search time every 20 seconds. The search times of P-(a) and P-(b) become shorter gradually until 250 seconds. However, along with the increase of the occurrence of Query messages from 300 seconds, the search times become longer.

5.2.3 The Number of Index Nodes

Figure 11 shows the number of index nodes every 20 seconds. In P-(b), the index nodes are allocated adaptively to the occurrence of Query messages. On the other hand, in P-(a), the number of index nodes drastically decreases and then repeatedly increases and decreases, after frequency of sending Query message quickly decreases from 300 seconds.

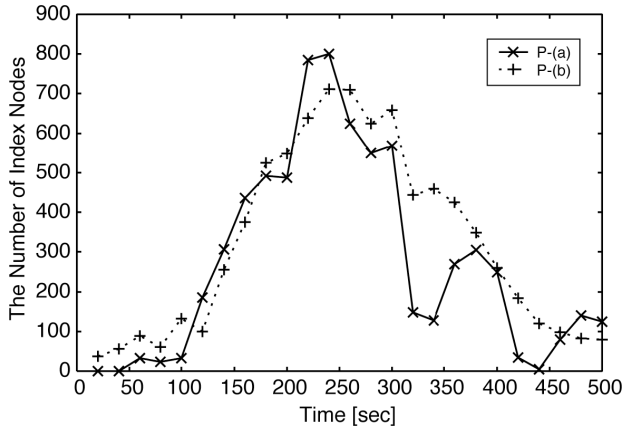

Fig. 11 The number of index nodes.

5.3 Discussion

In P-(a) while frequency of sending Query messages increases, many index nodes are constructed. However, while frequency of sending Query messages decreases, the number of index nodes quickly decreases. After that, increase and decrease of the number of index nodes are repeated. This phenomenon can be explained as follows.

Due to quick decrease of Query messages, many index nodes are suddenly destructed. To compensate excessive decrease of index nodes, some new index nodes are constructed. Such destruction and construction of index nodes are repeated until frequency of sending Query messages does not decrease. This phenomenon causes a temporal long search time. P-(a) is thus sensitive to quick decrease of frequency of sending Query messages.

In P-(b) while frequency of sending Query messages increases, nodes through which not only Query messages but also QueryHit messages are delivered make indices. These index nodes form a kind of the backbone networks. Even when frequency of sending Query messages quickly decreases, frequency of QueryHit messages does not decrease so much in the backbone network. As a result, the number of index nodes is gradually decreased. So that, P-(b) is stable to decrease frequency of sending Query messages. Therefore, search time by P-(b) is shorter than that by P-(a) even when quick decrease of Query message occurs.

6. Simulation Experiment 3

In simulation experiment 2, comparison between P-(a) and P-(b) was explained. In order to show an effect of network topology and frequency of in/out of nodes are explained in this section.

6.1 Simulation Models and Result

For frequency of in/out of node, we also performed three simulations, frequency=1 second, frequency=2 second and frequency=3 second, for four schemes, P-(a), P-(b),

Table 7 The result of simulation for three different frequencies (f).

f	Search Time (ms)				The number of messages			
	P-(a)	P-(b)	Gnutella	LI	P-(a)	P-(b)	Gnutella	LI
1	74	72	99	80	3,314,948	3,312,612	3,362,747	3,290,225
2	76	73	100	80	3,230,806	3,223,559	3,279,800	3,211,364
3	76	73	99	79	3,222,584	3,205,096	3,260,821	3,185,758

f	The number of entries				Hit ratio (%)			
	P-(a)	P-(b)	Gnutella	LI	P-(a)	P-(b)	Gnutella	LI
1	81,552	82,813	0	143,499	96.2	96.1	93.7	98.0
2	63,241	54,873	0	148,072	96.7	97.1	94.8	98.8
3	54,940	52,221	0	149,277	96.5	96.6	94.9	98.5

Table 8 The result of simulation for three different network topology (r).

r	Search Time (ms)			The number of messages		
	P-(a)	P-(b)	LI	P-(a)	P-(b)	LI
1	74	72	80	3,314,948	3,312,612	3,290,225
2	58	54	72	3,450,884	3,541,426	3,273,758
3	46	39	60	4,304,417	5,011,148	3,186,171

r	The number of entries			Hit ratio (%)		
	P-(a)	P-(b)	LI	P-(a)	P-(b)	LI
1	81,552	82,813	143,499	96.2	96.1	98.0
2	302,188	328,275	163,119	97.5	98.0	98.3
3	801,489	858,998	215,970	97.8	98.4	98.4

Gnutella and Local Indices scheme. The result is described in Table 7. This simulation was performed under the condition of r=1. For network topology, we performed three simulations, r=1, r=2 and r=3, for three schemes, P-(a), P-(b) and Local Indices scheme. The result is described in Table 8.

6.2 Discussion

As shown in Table 7, there was no significant difference when changing frequency except the number of entries for P-(a) and P-(b). When a node comes in and out, indices are changed. It is natural that if in/out interval becomes longer, the number of entries is lower. Also there was no significant difference between P-(a) and P-(b). This result means that both proposed schemes are stable for changing frequency. As shown in Table 8, proposes schemes show better search time comparing to Local Indices scheme. When r = 3, however, the number of messages and the number of entries of Local Indices scheme show better performance comparing to proposed schemes. When r is larger, index node in our proposed schemes receives more Report message, so that the number of messages and the number of entries become larger. When network topology was changed and the value of r became large, search time became shorter. For example, search time for P-(b) as r = 3 was faster about 180%. This means that gathering indices from wider area provides higher quality of pointers for information. However, the number of messages and entries become large, so the load for index node becomes large. The optimal size of r for proposed schemes is further study. P-(b) provided shorter search time comparing to P-(a), it means that deciding index node by using ratio of the number of QueryHit and Query is better mechanism.

7. Conclusions

This paper has proposed a new dynamic index allocation scheme for data retrieval and provision in P2P networks. According to change of the numbers of Query messages and QueryHit messages, index nodes are dynamically constructed and destructed. Also it is explained that dependability can be introduced by our scheme. The simulation results show that the number of messages and search time by the proposed schemes are much shorter than the previous schemes such as Gnutella and Local Indices. The performance of the proposed scheme can be maintained even when quick decrease of Query messages occurs. The proposed scheme therefore attains an efficient data retrieval and provision while avoiding waste of resources such as index entries and Query messages. By using our scheme, we proposed an reliable data retrieval technique in [15]. In the future research, we would like to refine our dynamic index allocation scheme to meet failure situation and to implement the proposed scheme to show the efficiency in the real P2P networks.

Acknowledgments

This work was supported in part by the Ministry of Educa-

tion, Science, Sports and Culture of Japan under Grant-in-Aid for Scientific Research (C) (No.15500049) and Young Scientists (B) (No.16700075), the Ministry of Internal Affairs and Communications of Japan under Grant-in-Aid for SCOPE-C (No.052308002) and Hiroshima City University under their research grants.

References

[1] Freenet website. http://www.freenet.com
[2] Gnutella website. http://www.gnutella.com
[3] Napster website. http://www.napster.com
[4] The network simulator—ns-2. Project web page available at http://www.isi.edu/nsnam/ns/
[5] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," Proc. Middleware2001, vol.2218 of Lecture Notes Comp. Sci., pp.329–350, Springer-Verlag, Berlin, 2001.
[6] T. Ohta, Y. Masuda, K. Mitsukawa, Y. Kakuda, and A. Ito, "A dynamic index allocation scheme for peer-to-peer networks," Proc. 7th IEEE International Symposium on Autonomous Decentralized Systems (ISADS2005) Workshops, AHSP2005 (1st International Workshop on Ad Hoc, Sensor and P2P Networks), pp.667–672, 2005.
[7] S. Saroiu, P.K. Gummadi, and S.D. Gribble, "A measurement study of peer-to-peer file sharing systems," Proc. 9th SPIE Conference on Multimedia Computing and Networking (MMCN2002), pp.407–418, 2002.
[8] S. Ratnasamy, P. Francis, M. Handley, and R. Karp, "A scalable content-addressable network," Proc. ACM SIGCOMM, pp.161–172, 2001.
[9] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, E. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," IEEE/ACM Trans. Netw., vol.11, no.1, pp.17–32, 2003.
[10] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," Proc. 22th IEEE Int'l. Conf. on Distributed Computing Systems (ICDCS 2002), pp.5–14, 2002.
[11] A. Crespo and H. Garcia-Molina, "Semantic overlay networks for p2p systems," Technical Report, Computer Science Department, Stanford University, Oct. 2002.
[12] A. Crespo and H. Garcia-Molina, "Routing indices for peer-to-peer systems," Proc. 22nd International Conference on Distributed Computing Systems (ICDCS'02), pp.23–32, 2002.
[13] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shanker, "Making gnutella-like P2P systems scalable," Proc. SIGCOMM 2003, pp.407–418, 2003.
[14] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient content location using interest-based locality in peer-to-peer systems," Proc. IEEE INFOCOM 2003, pp.2166–2176, 2003.
[15] A. Ito, T. Ohta, K. Mitsukawa, and Y. Kakuda, "An adaptive index allocation scheme for reliable data retrieval and provision in peer-to-peer networks," Proc. 21st Annual ACM Symposium on Applied Computing (SAC 2006), pp.697–704, April 2006.
[16] SHARMAN NETWORKS LTD., KaZaA Media Disktop, 2001. http://www.kazaa.com/
[17] H. Sunaga, T. Hoshiai, S. Kamei, and S. Kimura, "Technical trends in P2P-based communications," IEICE Trans. Commun., vol.E87-B, no.10, pp.2831–2846, Oct. 2004.

Atsushi Ito received the B.E. and M.Sc. degrees from Nagoya University, Japan, in 1981 and 1983, respectively. He joined Kokusai Denshin Denwa Co., Ltd. (KDD). From 1985, he has been with Research and Development Laboratories, KDDI. From 1991 to 1992, he was a visiting researcher in CSLI (Center for the Study of Language and Information), Stanford University, U.S.A. Currently, he is a senior manager of Technology Planning & Development Division of KDDI Corporation. His research interests include ad hoc networks and application of IT for education, judicially system and medical informatics.

Tomoyuki Ohta received the B.E., M.E., and Ph.D. degrees from Hiroshima City University, Japan, in 1998, 2000 and 2006, respectively. He is currently a research associate in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University, since 2002. His current research interests include mobile communication systems. He is a member of IEEE (U.S.A) and ACM (U.S.A).

Kouichi Mitsukawa received the B.E. and M.E. degrees from Hiroshima City University, Japan, in 2003 and 2005, respectively. He is currently working at New Media Research Institute Co., Ltd. His research interests include peer-to-peer communication systems.

Yoshiaki Kakuda received the B.E., M.Sc., and Ph.D. degrees from Hiroshima University, Japan, in 1978, 1980 and 1983, respectively. From 1983 to 1991, he was with Research and Development Laboratories, Kokusai Denshin Denwa Co., Ltd. (KDD). He joined Osaka University from 1991 to 1998 as an Associate Professor. He is currently a Professor in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University, since 1998. His current research interests include network software engineering, assurance networks and mobile ad hoc networks. He is a member of IEEE (U.S.A) and IPSJ (Japan). He received the Telecom. System Technology Award from Telecommunications Advanced Foundation in 1992.