# Performance Analysis of the Superpeer-based Two-layer P2P Overlay Network with the CBF Strategy

Kenichi Watanabe†, Naohiro Hayashibara††, and Makoto Takizawa††
†Information Science, Graduate School of Advanced Science and Technology, Tokyo Denki University
††Department of Computers and Systems Engineering, Tokyo Denki University
Email: {nabe, haya, taki}@takilab.k.dendai.ac.jp

## Abstract

*Peer-to-peer (P2P) systems are widely used in various types of applications. In this paper, we evaluate the superpeer-based two-layer (SBTL) P2P overlay network with the charge-based flooding (CBF) algorithm to detect target peers which have target files, proposed as our previous work. The SBTL P2P overlay network is composed of two layers, normal peer and superpeer layers which include normal peers and superpeers, respectively. Multiple normal peers with some common properties, e.g. peers which have replicas of a file, are interconnected with a superpeer. A collection of a superpeer and normal peers is referred to as a cluster. In a cluster, a normal peer tries to find a target peer which has a target file by itself without help of its superpeer. If no target peer is detected in the cluster, the normal peer asks the superpeer to find the target peer. Then, the superpeer forwards the request message to other superpeers by using a type of flooding algorithm named the CBF algorithm at the superpeer layer. We evaluate the SBTL P2P model in terms of the number of messages exchanged among peers and communication load compared with other models.*

## 1. Introduction

Various types of peer-to-peer (P2P) applications and systems are developed and used for achieving some objectives such as file sharing, distributed storage, instant messaging, distributed computation, contents delivery service, cooperative work. In P2P systems, a huge number of computers are interconnected in a network lying on the top of underlying physical computer network, typically the IP network. That is why the network is called an *overlay* [1,7] network. In the paper [1], P2P overlay networks are categorized in terms of levels of network centralization and structure. In a broad sense, there are three types of P2P systems, i.e. *centralized*, *decentralized and unstructured*, and *decentralized and structured* ones. On the other hand, in a narrow sense, there are five types of P2P systems, i.e. *centralized* [10], *decentralized and unstructured* [14, 19], *optimized decentralized and unstructured* [4, 5, 11, 12], *decentralized and highly structured* [13, 16, 18], and *decentralized and loosely structured* [2, 3] ones. Further discussion on categorization of P2P overlay networks can be found in the paper [1]. Since the superpeer models such as Kazaa [5] or Gnutella0.6 [4] are classified into an optimized decentralized and unstructured P2P system, we consider only optimized decentralized and unstructured P2P systems in this paper.

An optimized decentralized and unstructured P2P system such as Kazaa [5] is composed of *superpeers* and *normal* peers. A superpeer connected with some normal peers is equipped with sufficient CPU power, bandwidth, and storage capacity and plays a role of a controller of normal peers. For example, a superpeer manages index information of its normal peers and acts as a bridge/gateway between the normal peers and other superpeers and their normal peers. A normal peer has the same ability as the other normal peers have. That is, normal peers in an optimized decentralized and unstructured P2P system play the same role as peers (*servents*) in a decentralized and unstructured P2P system. If a new normal peer would like to join the system, the normal peer first sends its information to a superpeer and the superpeer adds the information to its index and manages the index for membership management and retrieval. Here, suppose a normal peer $p_{n_i}$ would like to find a target file. The normal peer $p_{n_i}$ sends a request message to the superpeer $p_s$. Then, the superpeer $p_s$ forwards the request message to a normal peer $p_{n_j}$ if the superpeer $p_s$ knows the normal peer $p_{n_j}$ holds the target file. Otherwise, the superpeer $p_s$ sends the request message to other superpeers connected with the superpeer $p_s$ in a flat unstructured overlay network. Decentralized and unstructured P2P systems are flooded with request messages which consume CPU resources and bandwidth of peers. On the other hand, in the superpeer models [4, 5, 11, 12], each superpeer efficiently manages its normal peers, their index information, and request messages. Therefore, the number of messages transmitted in an overlay network is reduced.

The authors propose a superpeer-based two-layer (SBTL) P2P overlay network with the charge-based flooding (CBF) algorithm in the paper [21]. The CBF [22] algorithm is a look-up protocol for distributed multimedia objects. An SBTL P2P overlay network is partitioned into clusters $C_{s_1}, \ldots, C_{s_m}$. Each cluster $C_s$ is composed of a superpeer $p_s$ and normal peers $p_{n_1}, \ldots, p_{n_m}$. At the normal peer layer, each normal peer $p_{n_i}$ is assumed to be in an *acquaintance* relation with every normal peer $p_{n_j}$ in $C_s$. A normal peer $p_{n_i}$ first tries to obtain a target file from another normal peer in $C_s$ without help of its superpeer $p_s$. If not found, $p_{n_i}$ asks its superpeer $p_s$ to obtain the target file. Then, the superpeer $p_s$ sends request messages to other superpeers at the superpeer layer by using the CBF algorithm. On behalf of time-to-live (TTL) [14, 19] and hops-to-live (HTL) [2] counters, a request message is first assigned with some amount of *charge* which shows the total number of request messages to be transmitted in the CBF algorithm. The charge of the request message is decremented by one each time the request message passes over a peer in a way similar to the flooding algorithm [2, 14, 19]. If there are multiple routes to target peers having a target file, the charge is divided into parts of charge which are assigned to a route based on the *ranking factor* and *trustworthiness* of the route. Here, the ranking factor [22,23] shows that how much a neighboring peer of a peer is trusted by other trustworthy neighboring peers of the peer, and the trustworthiness [22, 23] shows that how much each peer trusts its neighboring peers. In the CBF algorithm, a request message is rather forwarded to only peers which possess the potential of satisfying the request than broadcasting to all the superpeers.

It is easy to configure a P2P overlay network since it is not necessary to consider an underlying physical network. A route from a peer to another peer in an overlay network is realized on a physical route in the underlying physical network. However, a shorter route in an overlay network might show a longer physical route. Due to a *mismatch* between overlay and underlying networks, it takes longer time and consumes the bandwidth to propagate messages in the P2P overlay network [7, 15]. For instance, suppose a peer $p_T$ is in Tokyo, a peer $p_L$ is in Los Angeles, a peer $p_O$ is in Osaka, and a peer $p_N$ is in New York. The peer $p_T$ is connected to the peer $p_L$, $p_L$ is connected to the peer $p_O$, and $p_O$ is connected to the peer $p_N$ in an overlay network. In order for the peer $p_N$ to receive a message from the peer $p_T$, the message is delivered across the Pacific Ocean three times. In the overlay network, if there are connections between $p_T$ and $p_O$ in Japan and between $p_L$ and $p_N$ in the US, the message is delivered across the Pacific Ocean only once. In the paper [21], a peer assigns more volume of charge to a route if the route more matches the underlying physical network. That is, a route which is less congested should be used to send a message. In this paper, we evaluate the SBTL P2P overlay network with the CBF algorithm, proposed in the paper [21].

The rest of this paper is organized as follows. Section 2 introduces some fundamental background information and related studies. Section 3 briefly shows the SBTL P2P model. In Section 4, we evaluate the SBTL P2P overlay network in terms of the number of messages and communication load of the physical network.

## 2. Related Studies

### 2.1. Superpeer models

Kazaa [5] is one of the widely-used P2P applications using superpeer topology. Since it is proprietary, there are no detailed documents concerned with the P2P application. In the web page [5], a node with the fastest Internet connections and the most powerful computers are automatically selected as a superpeer. Normal peers connected to a superpeer automatically upload to the superpeer a small list of files the normal peers share. If a normal peer would like to find a target file, the normal peer first sends a search request to a superpeer. The superpeer forwards the request to other superpeers. After a requested file in another normal peer is detected, the file in the normal peer is directly downloaded to the requesting normal peer, not via the superpeer. The amount of CPU resources which can be used by a superpeer is limited to 10 [%] of the total CPU power available.

Edutella [11, 12] is a schema-based P2P network using superpeer (HyperCuP [17]) topology to provide access to digital resources. Superpeers hold RDF-based SP/P-RIs (superpeer/peer routing indices) and SP/SP-RIs (superpeer/superpeer routing indices) which is metadata information of peers and extracts from and summaries of all local SP/P-RIs, respectively. Moreover, similarity-based clustering of peers is offered and queries can be efficiently routed.

Gnutella 0.6 [4] uses superpeer topology, offering a dynamic superpeer selection mechanism. In Gnutella 0.6, superpeers are interconnected with each other in an unstructured P2P overlay network and normal peers are connected to a superpeer which is randomly selected. Here, note that Gnutella 0.4 [14, 19] consists of a decentralized and unstructured P2P system.

Morpheus [9] offers multiple network compatibility with other P2P file-sharing applications such as Neonet Network, Gnutella, BitTorrent, G2 and so forth. Grokster was also a P2P file-sharing application until November 7, 2005.

In these systems, a superpeer suffers from heavy workload, i.e. performance bottleneck, and becomes a single point of failure. If a superpeer fails or leaves a network, its normal peers are disconnected from the network until a new superpeer is selected. In the paper [24], fundamental characteristics of superpeer models are discussed and $k$-redundant superpeer topology is proposed to provide reliability and decrease the load on superpeers. In the paper [21], the authors assume that each normal peer can communicate with not only its superpeer but also other normal peers in a cluster so that the superpeer can be less loaded by reducing the communication overheads via the superpeer.

### 2.2. Matching overlay networks

The paper [15] shows only two to five percentage of Gnutella connections link nodes within a group of a local area network and more than forty percentage of all Gnutella nodes are located within the top 10 groups. This means most traffic generated by Gnutella crosses group borders and unnecessarily increases costs. If the overlay network topology is not appropriate to the physical topology, larger traffic is implied in the physical network and inappropriate overlay network topology increases costs for Internet service providers. Therefore, the paper also discusses how well the Gnutella overlay network topology maps to the physical Internet infrastructure.

Liu et al. propose AOTO (Adaptive Overlay Topology Optimization) [8] and a location-aware topology matching scheme [6] in which each peer issues a detector message in a small region so that other peers receiving the detector message can record relative delay information from the peer. Moreover, to improve the studies, the authors also discuss ACE (Adaptive Connection Establishment) [7] which builds an overlay multicast tree among a source peer and peers within a certain diameter from the source peer.

In our previous work [21], "matching" indicates how efficiently a request message can be delivered to the destination peer in the underlying physical network without reconstructing overlay network topology. That is, a route whose communication load is low is selected to delivery a request message. We discuss how to deliver request messages from a requesting peer to a target peer based on the physical network. A route with minimum load is selected and request messages are delivered on the route. Here, load of a route is measured by using ICMP (Internet Control Message Protocol) echo request and reply messages.

## 3. Superpeer-based Two-layer (SBTL) Overlay Networks

### 3.1. Clusters

A P2P overlay network is constructed on the underlying physical network. A SBTL P2P overlay network [21] consists of two layers, *normal peer* (lower) and *superpeer* (higher) *layers* [Figure 1]. The normal peer and superpeer layers are composed of a set of normal peers and a set of superpeers, respectively. Each normal peer is connected to a superpeer. Here, the normal peer is referred to as *acquainted* one of the superpeer. A collection of a superpeer $p_s$ and its acquainted normal peers $p_{n_1}, \ldots, p_{n_m}$ ($m \geq 1$) is referred to as a *cluster* $C_s$ [Figure 1]. A superpeer $p_{s_i}$ is connected with another superpeer $p_{s_j}$ at the superpeer layer.

### 3.2. Normal peer layer

Each normal peer $p_{n_i}$ can only communicate with a superpeer $p_s$ and acquainted normal peers in a cluster. Here, each normal peer is not directly communicated with normal peers and a superpeer in another cluster. That is, normal peers know what file every other normal peer holds and can trade a series of back-and-forth messages with each other within one hop, i.e. cost for exchanging a message is $O(1)$. To make this assumption effective, normal peers which have a common index or similar file can be grouped in a cluster. Because of this assumption, the number of communication between a superpeer and its normal peers can be reduced and the superpeer is lightly loaded.

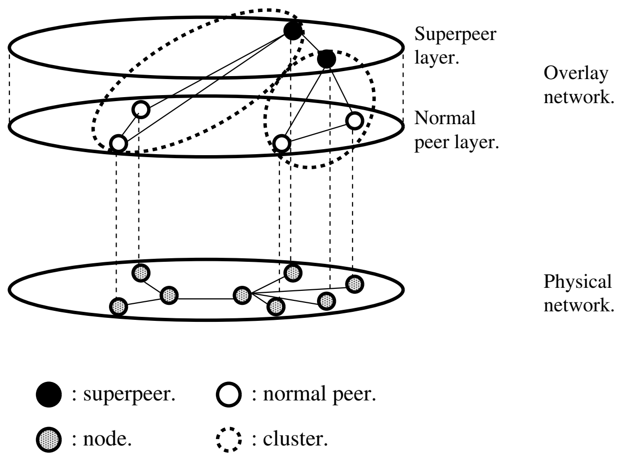

**Figure 1. SBTL overlay network.**

In the initial state, a requesting normal peer $p_n$ first sends a request message to its superpeer $p_s$ since the normal peer $p_n$ does not have acquaintance information. The superpeer $p_s$ returns the result with acquaintance information to the normal peer $p_n$. Here, the acquaintance information is controlled by valid time $t$ for checking if the acquaintance information is not obsolete. After the first retrieval, when the normal peer $p_n$ tries to find a target file, the normal peer $p_n$ checks if the valid time $t$ of the acquaintance information does not run out and there are acquainted normal peers which have the target file in its acquaintance table. If the valid time $t$ of the acquaintance information does not run and such acquainted normal peers are found in the acquaintance table of $p_n$, the normal peer $p_n$ directly accesses the acquainted normal peers to get the target file. Otherwise, for retrieving the target file and getting new acquaintance information, the normal peer $p_n$ asks its superpeer $p_s$ to find the target file.

A normal peer $p_n$ whose acquaintance information has been changed *pushes* new acquaintance information to its superpeer $p_s$. A normal peer $p_n$ does not push its acquaintance information periodically so that network traffic can be reduced. If the superpeer $p_s$ has new acquaintance information on other normal peers, the superpeer $p_s$ sends the acquaintance information to the normal peer $p_n$. Otherwise, the superpeer $p_s$ sends an $ACK$ (acknowledgment) message to the normal peer $p_n$. The other normal peers of the superpeer $p_s$ attempt to *pull* acquaintance information from the superpeer $p_s$ if necessary.

### 3.3. Superpeer layer

At the superpeer layer, a superpeer is connected with other superpeers in a flat unstructured overlay network. In the paper [21], in order to conform an overlay network to an underlying physical network, we define the communication load $d_{s_i,s_j}$, a parameter indicating a half of round-trip time from a superpeer $p_{s_i}$ to a superpeer $p_{s_j}$ which receives a request message. The communication load $d_{s_i,s_j}$ is defined as $((|msg_{s_i}|/b_{s_i} + \delta_{s_i}) + (|msg_{s_j}|/b_{s_j} + \delta_{s_j}) + PT_{s_j})/2$ where $|msg_{s_k}|$ [bit] indicates the length of a message $msg_{s_k}$, $b_{s_k}$ [bps] indicates the bandwidth of a superpeer $p_{s_k}$, $\delta_{s_k}$ [sec] indicates delay time ($k$ = $i$, $j$), and $PT_{s_j}$ indicates processing time of the superpeer $p_{s_j}$. To obtain the communication load, we take a *ping-style* strategy where ICMP echo request and reply messages are used. Not only the communication load is measured but also acquaintance information is exchanged by using the messages. More detailed algorithm of how to exchange acquaintance information is discussed in the paper [21].

For retrieval of a target file, there are two phases, *look-up* and *file transmission* phases. In the look-up phase, a normal peer $p_n$ which would like to obtain a target file checks if its acquainted normal peers have the target file. If at least one acquainted normal peer holds the target file, the normal peer $p_n$ tries to access the acquainted normal peer as discussed in the preceding subsection. Otherwise, the normal peer $p_n$ sends a request message to its superpeer $p_s$. Here, the superpeer $p_s$ starts propagating a request message to its superpeers by using the CBF [22] algorithm. On behalf of the TTL/HTL counter, a request message $rmsg$ which a superpeer $p_s$ issues is assigned with some integer value $V$, $rmsg.charge$ := $V$. The initial value of the charge $charge$ is decided based on the number of messages transmitted in the TTL or HTL based flooding algorithm. Here, suppose if the superpeer $p_s$ sends the request message $rmsg$ to other superpeers $p_{s_1}, \ldots, p_{s_m}$ and there are multiple routes to the destination superpeers, $rmsg.charge$ is divided into a certain amount of the charge based on the communication load discussed above, i.e. $rmsg_i.charge$ := $rmsg.charge$ × ((1 / $d_{s,s_i}$) / (1 / $d_{s,s_1}$ + $\cdots$ + 1 / $d_{s,s_m}$)). The superpeer $p_s$ sends the request message $rmsg_i$ with the $charge$ value to another superpeer $p_{s_i}$. Then, $rmsg_i.charge$ is decremented by one, $rmsg_i.charge$ := $rmsg_i.charge$ - 1 on receipt of $rmsg_i$.

The next step is the file transmission phase. Although source routing methods for forwarding a request message are inefficient, a target file could be transmitted by the source routing. In this phase, based on the communication load $d$, the route of file transmission is efficiently selected by a superpeer which has a target file. More detailed algorithm of how to efficiently transmit a target file is discussed in the paper [21].

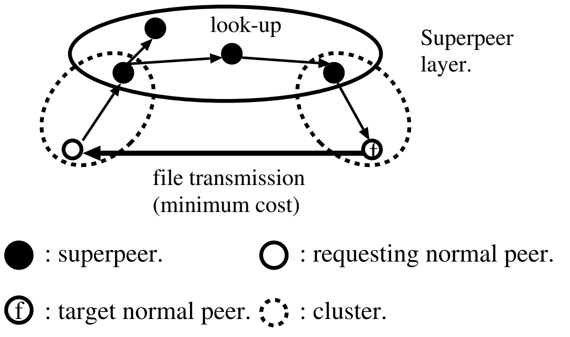

Figure 2. Look-up and file transmission.

## 4. Evaluation

### 4.1. Neko

We take usage of distributed simulator Neko [20] to evaluate our algorithm. Neko provides a uniform and extensible environment for the various phases of algorithm design and performance evaluation. The architecture of Neko is composed of *application* layers and a *network* layer. At the level of each layer in a hierarchy of application layers, each process $proc_i$ ($0 \leq i \leq n-1$) communicates with another process through a message passing interface. There are two types of application layers, passive and active ones. No thread is in the passive layer. At the layer below the passive layer, its $deliver$ method is invoked to push a message upon the passive layer. The active layer derived from class $NekoThread$ has its own thread. It actively pulls a message from the layer below the active layer by using its $receive$ method which blocks until a message arrives. When estimated time elapses, the method is called in a non-blocking mode. To interact with the layer below the active layer, it has to bypass the FIFO (First-In First-Out) message queue by providing its own $deliver$ method.

Each process of a distributed application has a $NekoProcess$ object placed between the application layers and network layer. The $NekoProcess$ takes three important roles:

1. The $NekoProcess$ holds some process wide information, e.g. the address of the process and the local time.
2. Some generally useful services such as logging messages are implemented.
3. The $NekoProcess$ dispatches and collects messages to the appropriate network if an application uses several networks in parallel.

The network layer can be instantiated from a collection of predefined networks and be in parallel managed with the networks by Neko.

### 4.2. Simulation setup

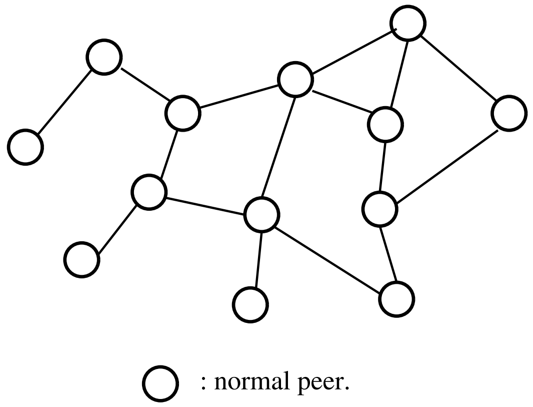

Figure 3. Unstructured overlay network (UON).

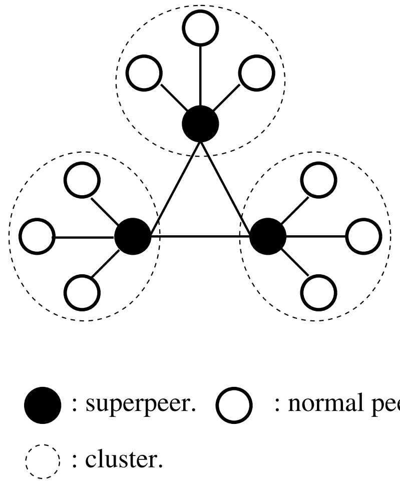

**Figure 4. Traditional superpeer model (TSPM).**

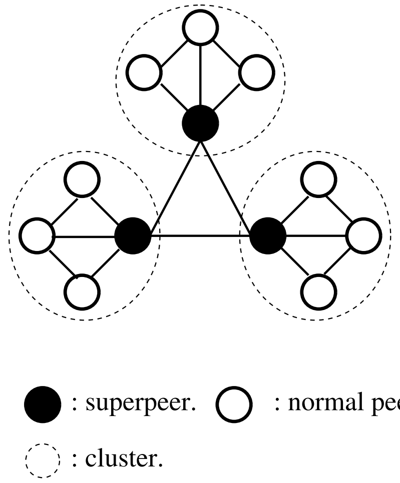

**Figure 5. Ring-topology overlay network (RTON).**

In this evaluation, we consider four types of network topology in order to evaluate our algorithm in terms of the number of transmitted messages and communication load. The first network topology is composed of an unstructured overlay network, abbreviated as UON, such as Gnutella 0.4 [14, 19] [Figure 3]. The second network topology is composed of a traditional superpeer model, abbreviated as TSPM, such as Kazaa [5] and Gnutella 0.6 [4] [Figure 4]. The third network topology is composed of a ring-topology overlay network abbreviated as RTON [Figure 5]. The fourth network topology is composed of the superpeer-based two-layer P2P overlay network, abbreviated as SBTL, which we discuss in the paper [21] [Figure 6].

Table 1 and 2 show simulation parameters used in this evaluation and their values. Let $N_P$, $N_{SP}$, and $N_{NP}$ mean the number of peers, superpeers, and normal peers in each overlay network, respectively. In UON [Table 1], there is no superpeer and each peer has the same ability, i.e. $N_P = N_{NP}$. Each peer has three neighboring peers randomly selected and a link between peers may be asymmetric. In TSPM, RTON, and SBTL [Table 2], since there are two roles, a superpeer and a normal peer, in each network topology, the number of peers in the network topology is equal to the number of superpeers and normal peers in the network topology ($N_P = N_{SP} + N_{NP}$). Each cluster has only one superpeer, i.e. the number of clusters is equal to the number of superpeers ($N_C = N_{SP}$). The cluster size $C_{SIZE}$ shows the number of a superpeer and normal peers in a cluster and is defined as $N_{SP}/N_C + N_{NP}/N_C$ = $(N_{SP} + N_{NP})/N_C$ = $N_P/N_C$. Here, the number of clusters multiplied by the cluster size shows the number of peers in the network topology ($N_P = N_C \times C_{SIZE}$). In TSPM, each superpeer has nine normal peers in a cluster ($N_{NP}/N_C$).

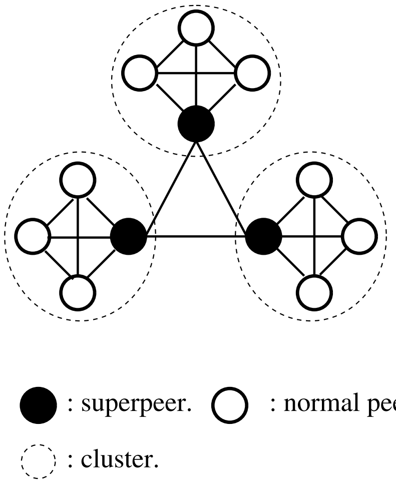

**Figure 6. Superpeer-based two-layer P2P overlay network (SBTL).**

**Table 1. Simulation parameters for UON.**

| Parameters | Assigned values |
|---|---|
| Number of peers $N_P$ | 1,000 |
| Number of normal peers $N_{NP}$ | 1,000 |

**Table 2. Simulation parameters for TSPM, RTON, and SBTL.**

| Parameters | Assigned values |
|---|---|
| Number of peers $N_P$ | 1,000 |
| Number of superpeers $N_{SP}$ | 100 |
| Number of normal peers $N_{NP}$ | 900 |
| Number of clusters $N_C$ | 100 |
| Cluster size $C_{SIZE}$ | 100 |

**Table 3. Number of objects in each overlay network.**

| Parameters | Assigned values |
|---|---|
| Number of files $N_F$ | 10 |
| Number of file types | 10 |

A link between a superpeer and a normal peer is symmetric. A link between superpeers may be asymmetric because superpeers neighboring to a superpeer are randomly selected. Moreover, there is no connection between normal peers. In RTON, each superpeer has nine normal peers in a cluster as well as TSPM. A link between a superpeer and a normal peer is symmetric and a link between superpeers may be asymmetric. Each normal peer knows both adjacent normal peers in a cluster. In SBTL, each superpeer has nine normal peers in a cluster as well as TSPM and RTON. A link between a superpeer and a normal peer is symmetric and a link between superpeers may be asymmetric. Each normal peer knows the other eight normal peers in a cluster and the connections are symmetric. Here, in each type of network topology, the communication load $d$ between peers is randomly decided based on studies in the paper [21].

As shown in Table 3, in each type of network topology, the number $N_F$ of files are distributed to normal peers. Here, files are distributed according to Zipf's low, $f(k; s, N_F) = 1/(k^s \cdot \sum_{n_f=1}^{N_F} 1/n_f^s)$ ($\sum_{k=1}^{N_F} f(k; s, N_F)$ = 1) where $N_F$ is the total number of files, $k$ is their rank, and $s$ is the exponent characterizing the distribution. Here, since we take the classic version of Zipf's low, $s$ = 1, i.e. $f(k; s, N_F) = 1/(k \cdot \sum_{n_f=1}^{N_F} 1/n_f)$. For instance, the number of the most popular file is $N_F/(k \cdot \sum_{n_f=1}^{N_F} 1/n_f)$ = 10 / ($\sum_{n_f=1}^{10} 1/n_f$) = 10 / 2.929 = 3.414 and the number of the most unpopular file is $N_F/(k \cdot \sum_{n_f=1}^{N_F} 1/n_f)$ = 10 / $(10 \cdot \sum_{n_f=1}^{10} 1/n_f)$ = 10 / 29.290 = 0.341.

## 4.3. Evaluation results

In this evaluation, we consider two evaluation scenarios. In the first scenario, the number of messages and communication load for membership management in each overlay network are measured. In UON, each peer sends a ping message with its acquaintance information to three neighboring peers every time unit and the neighboring peers return pong messages with their acquaintance information. In TSPM, each normal peer sends its acquaintance information to a superpeer every time unit. A superpeer sends its acquaintance information to three neighboring superpeers every time unit and the three neighboring superpeers return reply messages. In RTON, each normal peer sends its acquaintance information to a superpeer and both adjacent normal peers every time unit. Then, the two normal peers return reply messages. Superpeers behave in the same way as TSPM. In SBTL, one normal peer sends its acquaintance information to a superpeer every time unit and the other normal peers pull acquaintance information every five time units. Superpeers at the superpeer layer behave in the same way as TSPM.

Figure 7 shows the number of messages exchanged for membership management. The horizontal and vertical axes show a time unit number and the number of exchanged messages, respectively. From the figure, the number of exchanged messages in TSPM, RTON, and SBTL is fewer than the number of exchanged messages in UON since messages sent out of a cluster are exchanged through superpeers. Except the number of pull messages, the number of transmitted messages in SBTL is equal to the number of transmitted messages in TSPM. That is, compared with TSPM, the number of transmitted messages in SBTL depends on the pull messages. Figure 8 shows total communication load for membership management and indicates a similar result to Figure 7.

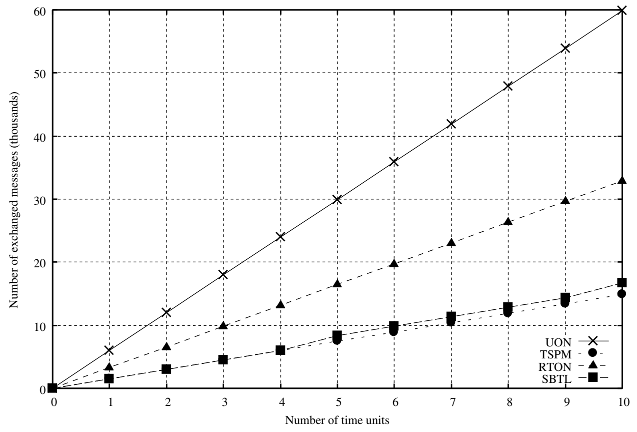

**Figure 7. Number of messages for membership management.**

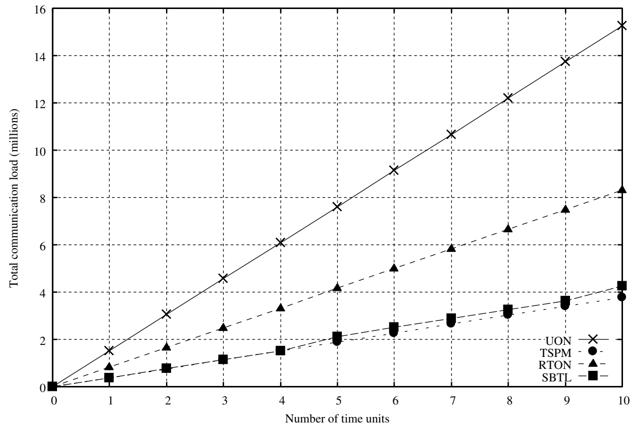

**Figure 8. Total communication load for membership management.**

In the second scenario, the number of messages and communication load for look-up of a target file are measured.

In UON, a requesting peer sends a request message to three neighboring peers. If the peers hold a target file, the peers return a positive reply to the requesting peer and the request message is not forwarded anymore. Otherwise, the peers forward the request message to their neighboring peers. Here, an initial TTL value of the request message is 7. In TSPM, a requesting normal peer first sends a request message to its superpeer in a cluster. If a normal peer which holds a target file exists in the cluster, the superpeer forwards the request message to the normal peer. Otherwise, the superpeer forwards the request message to three neighboring superpeers at the superpeer layer as well as UON. In RTON, a requesting normal peer first checks if both adjacent normal peers have a target file. If the normal peers hold the target file, the requesting normal peer directly sends a request message to the normal peers. Otherwise, the requesting normal peer sends the request message to its superpeer. The superpeer behave in the same way as TSPM. In SBTL, a requesting normal peer first checks if other eight normal peers hold a target object. If the normal peers hold the target file, the requesting normal peer directly sends a request message to the normal peers. Otherwise, the requesting normal peer sends the request message to its superpeer. The superpeer behave in the same way as TSPM.

Figure 9 shows the number of messages for look-up of a target file. The horizontal and vertical axes show a TTL value and the number of exchanged messages, respectively. The number of exchanged messages in SBTL is almost linear-proportional to a TTL value while the number of exchanged messages in UON, TSPM, and RTON grows exponentially based on a TTL value. Figure 10 shows total communication load for look-up and indicates a similar result to Figure 9. Table 4 shows hit ratio in each overlay network. The hit ratio shows how many target files each request message can find, i.e. the number of retrieved files divided by the number of exchanged messages shows the hit ratio. From the table, a target file can be more often detected in SBTL than UON, TSPM, and RTON.

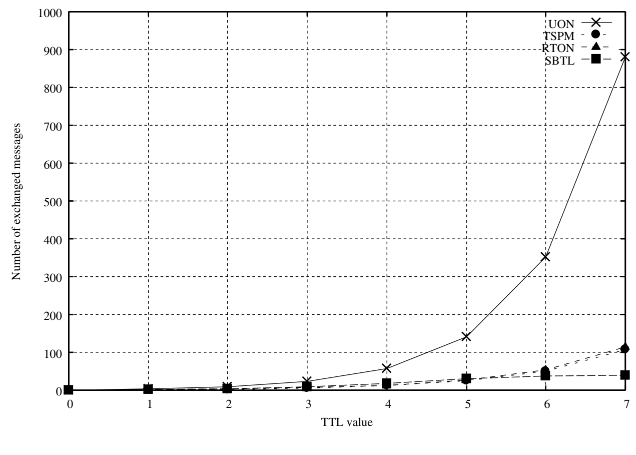

Figure 9. Number of messages for look-up.

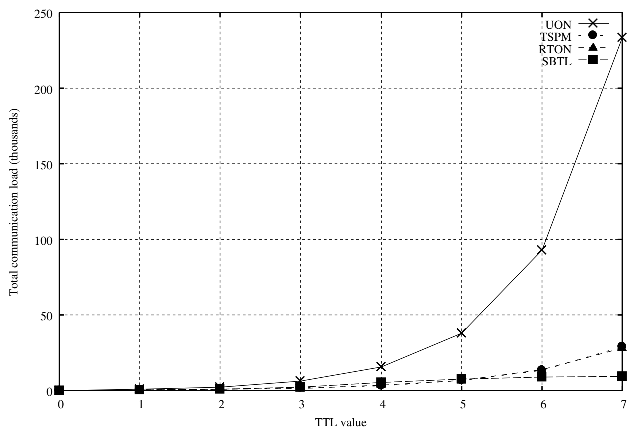

Figure 10. Total communication load for look-up.

Table 4. Hit ratio.

| Network topology | Hit ratio |
|---|---|
| UON | 0.0097 |
| TSPM | 0.1016 |
| RTON | 0.1538 |
| SBTL | 0.2017 |

## 5. Conclusions

In this paper, we evaluated the superpeer-based two-layer (SBTL) peer-to-peer (P2P) overlay network with the charge-based flooding (CBF) [22] algorithm [21]. In the SBTL P2P overlay network, each normal peer can directly communicate with every other normal peer in a cluster while each normal peer can only communicate with a superpeer in traditional superpeer models. A superpeer communicates with other superpeers to deliver a request message to target clusters by using the CBF algorithm. In the evaluation, we showed the number of messages exchanged and communication load can be reduced in our SBTL P2P model compared with other models.

## Acknowledgment

This research is partially supported by Research Institute for Science and Technology [Q06J-07] and Frontier Research and Development Center [18-J-6], Tokyo Denki University.

## References

[1] S. Androutsellis-Theotokis and D. Spinellis, “A Survey of Peer-to-Peer Content Distribution Technologies,” *ACM Computing Surveys*, 36(4):335–371, Dec. 2004.

[2] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, “Freenet: A Distributed Anonymous Information Stor-

age and Retrieval System," *Proceedings of the workshop on Design Issues in Anonymity and Unobservability (DIAU2000)*, pages 46–66, Jul. 2000.

[3] Freenet, http://www.freenetproject.org/.

[4] Gnutella - A Protocol for a Revolution, http://rfc-gnutella.sourceforge.net/.

[5] Kazaa, http://www.kazaa.com/.

[6] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang, "Location-Aware Topology Matching in P2P Systems," *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM2004)*, 2004.

[7] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "A Distributed Approach to Solving Overlay Mismatching Problem," *Proceedings of the 24th IEEE International Conference on Distributed Computing System (ICDCS2004)*, pages 132–139, Mar. 2004.

[8] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "AOTO: Adaptive Overlay Topology Optimization in Unstructured P2P Systems," *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM2003)*, pages 4186–4190, Dec. 2003.

[9] Morpheus, http://www.morpheus.com/.

[10] Napster, http://www.napster.com/.

[11] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch, "EDUTELLA: A P2P Networking Infrastructure Based on RDF," *Proceedings of the 11th International World Wide Web Conference (WWW2002)*, pages 604–615, May 2002.

[12] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. T. Schlosser, I. Brunkhorst, and A. Loser, "Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks," *Proceedings of the 12th International World Wide Web Conference (WWW2003)*, pages 536–543, May 2003.

[13] S. Ratnasamy, P. Francis, and S. Handley, "A Scalable Content-Addressable," *ACM SIGCOMM2001*, pages 161–172, Aug. 2001.

[14] M. Ripeanu, "Peer-to-Peer Architecture Case Study: Gnutella Network," *Proceedings of the International Conference on Peer-to-Peer Computing (P2P2001)*, pages 99–100, Aug. 2001.

[15] M. Ripeanu, I. Foster, "Mapping the Gnutella Network," *IEEE Internet Computing*, 6(1):50–57, Jan./Feb. 2002.

[16] A.I.T Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems," *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg (Middleware)*, pages 329–350, Nov. 2001.

[17] M. T. Schlosser, M. Sintek, S. Decker, and W. Nejdl, "HyperCuP–Hypercubes, Ontologies and Efficient Search on P2P Networks," *Proceedings of Agents and Peer-to-Peer Computing (AP2PC2002)*, pages 112–124, Jul. 2002.

[18] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," *IEEE/ACM Transactions on Networking (TON)*, 11(1):17–32, Feb. 2003.

[19] The Gnutella Protocol Specification v0.4, http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[20] P. Urban, X. Defago, and A. Schiper, "Neko: A Single Environment to Simulate and Prototype Distributed Algorithms," *Journal of Information Science and Engineering (JISE)*, 18(6):981–997, Nov. 2002.

[21] K. Watanabe, N. Hayashibara, and M. Takizawa, "A Superpeer-based Two-layer P2P Overlay Network with the CBF Strategy," to appear in *Proceedings of the 1st International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, Apr. 2007.

[22] K. Watanabe, N. Hayashibara, and M. Takizawa, "CBF: Look-up Protocol for Distributed Multimedia Objects in Peer-to-Peer Overlay Networks," *Journal of Interconnection Networks (JOIN)*, 6(3):323–344, Sep. 2005.

[23] K. Watanabe, M. Takizawa, "Service Oriented Cooperation among Trustworthy Peers," *Journal of Interconnection Networks (JOIN)*, 7(4):507–533, Dec. 2006.

[24] B. Yang and H. Garcia-Molina, "Designing a super-peer network," http://infolab.stanford.edu/ byang/pubs/superpeer.pdf.