P2Pオーバーレイネットワークにおけるピア間の信頼度

中島 嘉男 † 渡辺 健一 †‡ 林原 尚浩 † 滝沢 誠 † S.Misbah Deen◇
† 東京電機大学 理工学部 情報システム工学科
‡University of California, Irvine , USA
◇ University of Keele, England

Abstract

P2Pネットワーク内でオブジェクトがどのピアに存在しているかを発見する方法は議論されてきているが、オブジェクトにアクセス権がないと利用できない。アクセス権を持ったピアだけが、オブジェクトを操作することが出来る。本論文では、各ピアの知人ピアを用いた方法を提案する。アクセス権を考慮したピアの知人関係について議論する。次に、ピアがどの程度、各知人を信頼するかについて議論する。各知人ピアの信頼値を定義する。信頼値と知人の概念に基づいた電荷拡散 (CBF) アルゴリズムを示す。

Trustworthiness of Peers in Peer-to-Peer Overlay Networks

Yoshio Nakajima†, Kenichi Watanabe†‡, Naohiro Hayashibara†, Makoto Takizawa†, and S.Misbah Deen◇
†Tokyo Denki University, Japan
‡University of California, Irvine , USA
◇ University of Keele, England
E-mail {nak, nabe, haya, taki}@takilab.k.dendai.ac.jp, deen@cs.keele.ac.uk

An object is a unit of resource distributed in peer-to-peer (P2P) overlay networks. Service supported by an object is modeled to be a set of methods and quality of service (QoS). It is critical to discuss what peer can manipulate an object in what method, i.e. only a peer granted an access right can manipulate an object. In this paper, we take an acquaintance approach. An acquaintance of a peer p is a peer whose service the peer p knows and with which the peer p can directly communicate. We discuss types of acquaintance relations of peers with respect to what objects each peer holds and what access rights each peer is granted and can grant to another peer. Acquaintance peers of a peer may notify the peer of different information on target peers. Here, it is critical to discuss how much a peer trusts each acquaintance. We define the trustworthiness of an acquaintance in terms of the acquaintance relations among the peers. In addition, we present a charge-based flooding (CBF) algorithm to find target peers based on trustworthy acquaintances so that more trustworthy areas are more deeply searched.

1 Introduction

Various types and huge number of computers are interconnected in *peer-to-peer* (P2P) overlay networks [4]. An object is a unit of resource like database. An object is an encapsulation of data and methods for manipulating the data. A group of peer processes (abbreviated *peers*) on computers are cooperating by manipulating objects and exchanging messages in networks. Service supported by each object is characterized by types of methods. In a P2P overlay network, the huge number of peers are included and the membership of peers is dynamically changed. If a peer would like to obtain some service of an object, the peer has to find target peers which can manipulate the object in a required method.

Service supported by a peer is characterized by what objects the peer stores, what objects the peer can manipulate by what types of methods, and what access rights the peer can grant to other peers. An *acquaintance* of a peer p_i is another peer p_j whose service p_i perceives p_j to support for p_i. A peer first asks acquaintances to detect target peers which can manipulate a target object. Even if a peer holds a target object, the peer cannot be asked to manipulate the object if the peer is not granted an access right. Thus, even if peers with target objects are found, the objects cannot be manipulated without access rights on the target objects. If peers which satisfy the requirement on types of service and QoS are not detected, each acquaintance furthermore asks its acquaintances. Thus, access requests are propagated from acquaintances to acquaintances in a P2P overlay network. Acquaintance concepts are so for discussed in papers [2], but are used to just detect a target peer. In this paper, we discuss how to manipulate objects in addition to detecting where objects exist. We first define types of acquaintances are defined based on services, *holder* peers where objects are stored, *manipulation* peers which can manipulate objects, and *authorization* peers which can grant access rights.

If service supported by a peer is changed, the change information is distributed. However, it takes time to propagate the change to peers in the P2P overlay network. Hence, some acquaintances of a peer may show obsolete and inconsistent information on target peers of a target object. Hence, it is critical to discuss how much a peer trusts its acquaintance. A requesting peer is satisfiable for each access request to find a target peer if a target peer is detected. However, the requesting peer is not satisfiable for a manipulation request if the peer is not granted an access right. We define the satisfiability of each type of access right. Then, we define the *trustworthiness* of an acquaintance based on the satisfiability of each access request, i.e. by newly taking into account access rights.

In flooding algorithms [1, 8] to, counters like TTL (time-to-live) [8] and HTL (hops-to-live) [1] are used to prevent indefinite circulation and explosion of access request messages transmitted in networks. In this paper, we newly discuss a *charge-based flooding* (CBF) algorithm where a more trustworthy area is more deeply searched. An access request to a more trustworthy acquaintance is assigned with large amount of *charge* which shows the total number of messages to be transmitted.

In section 2, we discuss acquaintance relations of peers. In section 3, we discuss the trustworthiness of an acquaintance. In section 4, we discuss the CBF algorithm.

2 Acquaintances

2.1 Peer-to-object (P2O) relations

In peer-to-peer (P2P) overlay networks [1,5, 7–10, 14], it is discussed only how to detect a peer with a target object. Even if the location of a target object is detected, the object cannot be manipulated without an access right. An access right is specified in a form $[o, op]$ where o shows an object and op indicates a method of o. If a peer is granted an access right $[o, op]$, the peer can manipulate an object o in a method op. Hence, we discuss relations among peers and objects by taking into account access rights.

We have to find a target peer which supports some service on target objects and which is allowed to manipulate the objects. First, an application issues an access request $\langle o, op\rangle$ to a local peer p to manipulate a target object o with a method op. Here, the peer p is referred to as *initial* peer of the access request $\langle o, op\rangle$. A *target peer* of an access request is a peer which can support service satisfying the access right. For example, a target peer of $\langle o, op\rangle$ is a peer which manipulate a target object o by the method op. An object may be replicated in multiple peers. Hence, there might be multiple target peers of an access request $\langle o, op\rangle$ which can manipulate replicas of the object o through a method op.

On receipt of an access request $\langle o, op\rangle$, a peer has to find target peers of the access request. It is difficult, maybe impossible for each peer to perceive what service of what objects each peer supports due to the scalability. In addition, the type of service and quality of service (QoS) supported by each peer are dynamically changed.

Let **P** be a set of peers and **O** be a set of objects in a P2P overlay network. There are following types of peer-to-object (P2O) relations, $|$, $\models$, $\vdash$, and $\Box$ ($\subseteq \mathbf{P}\times\mathbf{O}$) for a peer p, an object o, and a method op [Figure 1] [12, 13]:

a. A peer p *holds* an object o ($p \mid o$) if the object o is stored in p. Here, p is a *holder* peer.
b. A peer p *can manipulate* an object o through a method op ($p \models_{op} o$), i.e. p is granted an access right $[o, op]$. Here, p is a *manipulation* peer.
c. A peer p *can grant* an access right $[o, op]$ to another peer p' ($p \vdash_{op} o$). Here, p is a *authorization* peer.
d. A peer p can do something for an object o by using a method op ($p \Box_{op} o$) iff $p \mid o$, $p \models_{op}$, or $p \vdash_{op} o$.

Even if a peer p holds an object o ($p \mid o$), p may not be granted an access right $[o, op]$ ($p \not\models_{op} o$). In the discretionary access control model [3], $p \vdash_{op} o$ iff $p \models_{op} o$. In the mandatary model, $p \vdash_{op} o$ may not hold even if $p \models_{op} o$. Relational database systems take the discretionary model [6, 11].

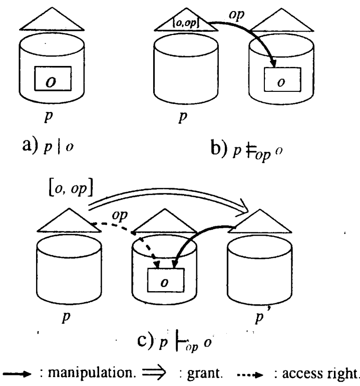

Figure 1. P2O relations.

The following types of peer-to-object (P2O) relations are defined for a peer p and an object o:

- $p \models o$ if $p \models_{op} o$ for some method op.
- $p \vdash o$ if $p \vdash_{op} o$ for some method op.
- $p \Box o$ iff $p \Box_{op} o$ for some method op.

In the discretionary model, a peer p can ask the authorization peer to grant an access right $[o, op]$ to the peer p if p is not granted the access right [Figure 2]. If p could not be granted an access right $[o, op]$, p can find another manipulation peer p' which is granted $[o, op]$ and asks p' to manipulate o. If p' agrees on manipulating o, p' manipulates o in another peer p'' on behalf of the requesting peer p as shown in Figure 3. Then, p' sends the result to p. Here, the manipulation peer p' is referred to as a *surrogate* of the peer p.

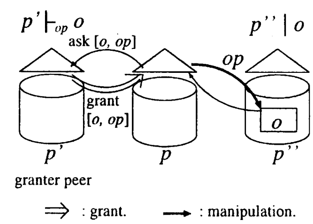

Figure 2. Authorization peer.

2.2 Acquaintance relations

Each peer cannot perceive where every object exists and how each object can be manipulated due to the scalability. Each peer obtains information on objects from the acquaintances. We discuss acquaintance relations among peers by using the peer-to-object (P2O) relations $|$, $\models$, and $\vdash$. Acquaintances of a peer p are peers whose service p knows, i.e. holder, manipulation, and authorization peers. For example, if a peer p knows that another peer p_i can manipulate

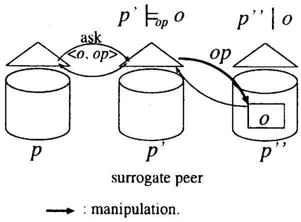

Figure 3. Surrogate peer.

an object o in a method op ($p_i \models_{op} o$), p_i is an acquaintance of p. If a peer p knows that another peer p_i has an acquaintance peer p_j, p_i is an acquaintance of p.

There are following types of acquaintance relations $\rightarrow$ for a peer p, an object o, and a method op:

- A relation "$p \rightarrow (p_i \mid o)$" holds iff p perceives that another peer p_i serves o ($p_i \mid o$) .
- $p \rightarrow (p_i \models_{op} o)$ iff p perceives that a peer p_i can manipulate o through op ($p_i \models_{op} o$) .
- $p \rightarrow (p_i \vdash_{op} o)$ iff a peer p perceives that a peer p_i can grant an access right $\langle o, op \rangle$ ($p_i \vdash_{op} o$).

If $p \rightarrow (p_i \mid o)$ and $p \models_{op} o$, a peer p can issue an access request $\langle o, op \rangle$ to another peer p_i because p not only knows where an object o is but also can manipulate o.

The following acquaintance relations are defined for an object o and a method op:

- $p \rightarrow (p_i \models o)$ if $p \rightarrow (p_i \models_{op} o)$ for some method op.
- $p \rightarrow (p_i \vdash o)$ if $p \rightarrow (p_i \vdash_{op} o)$ for some method op.
- $p \rightarrow (p_i \Box_{op} o)$ if $p \rightarrow (p_i \mid o)$, $p \rightarrow (p_i \models_{op} o)$, or $p \rightarrow (p_i \vdash_{op} o)$.
- $p \rightarrow^* (p_i \Box_{op} o)$ iff $p \rightarrow (p_i \Box_{op} o)$ or $p \rightarrow (p_k \rightarrow^* (p_i \Box_{op} o)$ for some peer p_k where $\Box \in \{ \mid, \models, \vdash \}$.
- $p \rightarrow^+ (p_i \Box_{op} o)$ iff $p \rightarrow (p_k \rightarrow^* (p_i \Box_{op} o))$ for some peer p_k.
- $p \rightarrow (p_i \Box o)$ iff $p \rightarrow (p_i \Box_{op} o)$ for op.
- $p \rightarrow^* (p_i \Box o)$ iff $p \rightarrow^* (p_i \Box_{op} o)$ for some op.
- $p \rightarrow^+ (p_i \Box o)$ iff $p \rightarrow^+ (p_i \Box_{op} o)$ for some op.

An *acquaintance* of a peer p_i is another peer p_j which knows where objects are stored, how objects can be manipulated, and what access rights p_j can grant to other peers. The following types of acquaintance relations $\Rightarrow_o^{\Box_{op}}$ $\Rightarrow_o^{\Box}$, $\Rightarrow_o$, and $\Rightarrow$ ($\subseteq \mathbf{P} \times \mathbf{P}$) are defined for a set $\mathbf{P}$ of peers:

- A peer p_j is an acquaintance of a peer p_i for an object o with respect to a method op and a P2O relation $\Box$ ($\in \{ \mid, \models, \vdash \}$) ($p_i \Rightarrow_o^{\Box_{op}} p_j$) if $p_i \rightarrow (p_j \Box_{op} o)$, p_i perceives "$p_k \rightarrow (p_j \Box_{op} o)$" for some p_k, or p_i perceives "$p_k \Rightarrow_o^{\Box_{op}} p_j$" for some p_k.
- $p_i \Rightarrow_o^{\Box} p_j$ iff $p_i \Rightarrow_o^{\Box_{op}} p_j$ for some method op.
- A peer p_j is an acquaintance of a peer p_i on an object o with respect to a method op ($p_i \Rightarrow_o^{op} p_j$) if $p_i \Rightarrow_o^{\Box_{op}} p_j$ for some P2O relation $\Box$.
- $p_i \Rightarrow_o p_j$ iff $p_i \Rightarrow_o^{op} p_j$ for some method op.
- A peer p_j is an *acquaintance* of a peer p_i ($p_i \Rightarrow p_j$) if $p_i \Rightarrow_o p_j$ for some object o.

If $p_i \Rightarrow_o^{\mid} p_j$, a peer p_j is perceived by p_i to be holder acquaintance of p_i with respect to an object o. If $p_i \Rightarrow_o^{\models} p_j$ and $p_i \Rightarrow_o^{\vdash} p_j$, p_j is as *manipulation* and *authorization* acquaintances of o, respectively. If $p_i \Rightarrow_o^{\Box} p_j$, $p_i \Rightarrow_o^{\Box} p_k$, $p_j \Rightarrow_o^{\Box} p_k$, and $p_j \not\Box o$, p_j is referred to as a *closer* acquaintance of p_i than another p_k with respect to o.

- $p_i \Rightarrow p_j$ (a peer p_j is an *acquaintance* peer of a peer p_i) iff $p_i \Rightarrow_o p_j$ for some object o.

The acquaintance relation amoung peers is reflexive but is neither symmetric nor transitive. Let *view*(p_i) be a set $\{ p_j \mid p_i \Rightarrow p_j \}$ of acquaintance peers of a peer p_i.

2.3 Cooperation of acquaintances

Suppose that a peer p_i would like to issue an access request $\langle o, op \rangle$ to a peer p_j. A manipulation peer which is not only granted an access right $[o, op]$ but also can manipulate o on behalf of another peer is a *surrogate* peer of o. If p_i perceives a surrogate peer p_j of a server p_k as an acquaintance ($p_i \Rightarrow p_j$), p_i can ask the surrogate p_j to make an access to an object o in p_k on behalf of p_i. Figure 4 shows the interaction among the peers p_i, p_j, and p_k.

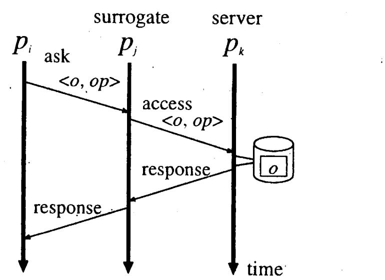

Figure 4. Surrogate peer.

Secondly, there is no surrogate peer for a peer p_i which issues an access request $\langle o, op \rangle$. The peer p_i first finds a authorization acquaintance on an object o. Here, p_j is found to be an authorization acquaintance of p_i ($p_i \Rightarrow_o^{\vdash} p_j$). Then, the peer p_i asks the authorization peer p_j to grant an access right $[o, op]$ to p_i. The requesting peer p_i manipulates o if p_i is granted $[o, op]$ as shown in Figure 5.

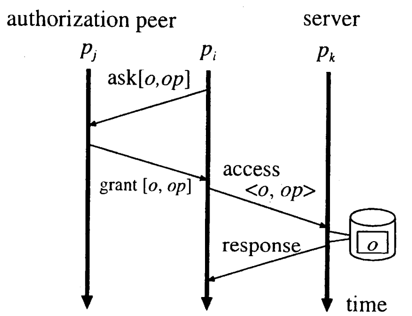

Figure 5. Authorization peer.

3 Trustworthiness

3.1 Satisfiability

There are multiple acquaintance peers on an object o for each peer p_i. The peer p_i has to find some acquaintance p_j which p_i can trust. We discuss how much each peer trusts an acquaintance. The peer p_i asks some acquaintance peer to manipulate a target object o in a method op. It is critical to discuss how much p_i trusts an acquaintance p_j. An access request $\langle o, \Box, op \rangle$ issued by p_i is specified in a tuple $\langle o, \Box, op \rangle$ for an object o, method op, and P2O relation $\Box$, which means as follows:

1. $\langle o, |, \_ \rangle$: a peer p_i would like to know in which peer an object o exists.
2. $\langle o, \models, op \rangle$: p_i would like to manipulate an object o through a method op.
3. $\langle o, \vdash, op \rangle$: p_i would like to be granted an access right $[o, op]$.

Suppose a peer p_i issues an access request $\langle o, \models, op \rangle$ to another peer p_j. There are two cases with respect to what p_j can do for the object: $p_j \models_{op} o$ (p_j can manipulate o through op) or $p_j \not\models_{op} o$. First, we suppose $p_j \models_{op} o$. If p_j holds the object o, i.e. $p_j \mid o$, p_j locally manipulates o in p_j. Then, p_j sends the reply r_i to p_i. Here, the requesting peer p_i is satisfied because p_i can obtain the result for the access request $\langle o, op \rangle$. Unless $p_j \mid o$, p_i or p_j has to detect a holder peer p_k of o. Here, p_i asks p_j to detect a holder peer of o. Suppose p_j finds a holder p_k of o in the acquaintances. The manipulation peer p_j issues an access request $\langle o, \models, op \rangle$ to p_k if $p_j \models_{op} o$. Here, p_i is less satisfiable since p_i cannot directly get the result from the acquaintance p_j. If the manipulation peer p_j does not agree on finding a holder peer of o, the requesting peer p_i finds a holder peer. If p_i detects a holder peer p_k ($p_k \mid o$), p_i informs the manipulation peer p_j of p_k.

Next, suppose that p_j just holdes an object, i.e. $p_j \mid o$ and p_i cannot manipulate o, i.e. $p_i \not\models_{op} o$. If p_i gets the access right $[o, op]$ from some authorization acquaintance, p_i can issue the method op to the holder peer p_j of the object o. The peer p_i finds an authorization acquaintance p_j of $[o, op]$. If found, p_i asks p_j to grant $[o, op]$. In another way, p_i finds a surrogate peer to manipulate the object o in p_j. If found, p_i asks p_j to manipulate o on behalf of p_i.

Suppose a peer p_i holds a object o ($p_i \mid o$) but is not granted an access right $[o, op]$ ($p_i \models_{op} o$). In the first way, the peer p_i finds agranted a acquaintance p_j on the access right $[o, op]$. If found, p_i asks p_j to grant $[o, op]$. In another way, the peer p_i finds a acquaintance p_j of an object o. If a manipulation acquaintance p_j is found, p_i asks p_j to manipulate the object o an behalf of p_i. If p_j agrees, p_j is a surrogate and manipulates o through a method op.

We define the *satisfiability* σ_{ij} of a peer p_i to another peer p_j in terms of type of request $\langle o, \Box, op \rangle$ and states of the peers p_i and p_j. Table 1 summarizes the satisfiability σ_{ij} for an access request $\langle o, \Box, op \rangle$ which a peer p_i issues to another peer p_j. Here, $0 < \delta \leq 1$. States of p_i and p_j show types of the service supported by p_i and p_j, respectively. For example, $p_j \mid o$ shows that p_j serves o. Here, if p_i issues an access request $\langle o, |, \_ \rangle$ to p_j, p_i finds p_j to hold the object o. The satisfiability $\sigma_{ij}(\langle o, |, \_ \rangle)$ is 1, i.e. the requesting peer p_i is satisfied since p_i can directly obtain the result of the access request $\langle o, |, \_ \rangle$ from the acquaintance peer p_j. Next, if a relation "$p_j \mid o$" does not hold ($p_j \not| o$) but the peer p_j knows another peer p_k is a holder of o, $p_j \rightarrow (p_k \mid o)$, p_i cannot get the result from p_j but may get the result from p_k. $\sigma_{ij}(\langle o, |, \_ \rangle)$ is defined to be δ. If a peer p_l is an acquaintance of p_i and $p_l \rightarrow (p_m \rightarrow (p_k \mid o))$, $\sigma_{il}(\langle o, |, \_ \rangle) = \delta \cdot \sigma_{lk}(\langle o, |, \_ \rangle) = \delta^2 \cdot \sigma_{mk}(\langle o, |, \_ \rangle) = \delta^3$. For an access request $\langle o, \models, op \rangle$, if p_i is granted an access right $[o, op]$ ($p_i \models_{op} o$) and knows that another peer p_j serves an object o ($p_j \mid o$), p_i obtains result by issuing a method op to the object o in the peer p_j. Hence, $\sigma_{ij}(\langle o, \models, op \rangle)$ is 1. In the paper, we assume δ to be 1/2.

3.2 Trustworthiness

Based on the satisfiability σ_{ij} $(\langle o, \Box, op \rangle)$ of an access request $\langle o, \Box, op \rangle$, a peer p_i makes a decision on how much p_i can trust an acquaintance p_j. The trustworthiness $\tau_{ij}(\langle o, \Box, op \rangle)$ from a peer p_i to another peer p_j with respect to an access request $\langle o, \Box, op \rangle$ is obtained on the basis of the satisfiability of access requests issued to the peer p_j. One idea is that the satisfiabilites of access requests issued to p_j are kept in record by the peer p_i. It is cumbersome to maintain the history of access requests and the satisfiabilites. Each time a peer p_i obtains the satisfiability $\sigma_{ij}(\langle o, \Box, op \rangle)$ from another peer p_j, the trustworthiness $\tau_{ij}(\langle o, \Box, op \rangle)$ is recalculated as follows:

$$\tau_{ij}(\langle o, \Box, op \rangle) := \alpha \cdot \tau_{ij}(\langle o, \Box, op \rangle) + (1-\alpha) \cdot \sigma_{ij}(\langle o, \Box, op \rangle).$$

Here, α is a constant ($0 \leq \alpha \leq 1$). The smaller α is, the more important the current request $\langle o, \Box, op \rangle$ is. If $\alpha = 1$, τ_{ij} is not changed. If $\alpha = 0$, the trustworthiness is decided only by the current satisfiability.

Suppose a peer p_i issues an access request $\langle o, \Box, op \rangle$ to another peer p_j. Here, p_j does not support $p_j \Box_{op} o$ but p_j perceives that some peer p_k supports the required service ($p_k \Box_{op} o$). The peer p_i obtains the satisfiability $\sigma_{ij}(\langle o, \Box, op \rangle)$ from Table 1. Then, p_j informs p_i of the P2O relation $p_k \Box_{op} o$. Then, the peer p_i issues an access request $\langle o, \Box, op \rangle$ to p_k. The peer p_i obtains the reply from p_k and the satisfiability $\sigma_{ik}(\langle o, \Box, op \rangle)$ [Figure 6]. If p_k is less satisfiable, $\tau_{ij}(\langle o, \Box, op \rangle)$ is decreased. τ_{ij} $(\langle o, \Box, op \rangle)$ is changed as follows:

$$\tau_{ij}(\langle o, \Box, op \rangle) := [\beta + (1 - \beta) \cdot \sigma_{ik}(\langle o, \Box, op \rangle)] \cdot \tau_{ij}(\langle o, \Box, op \rangle).$$

This means, the trustworthiness τ_{ij} $(\langle o, \Box, op \rangle)$ is decreased if a peer p_j introduces a less trustworthy peer p_k to a peer p_i. Here, $0 \leq \beta \leq 1$. The smaller β is, the more the satisfiability $\sigma_{ik}(\langle o, \Box, op \rangle)$ dominates $\tau_{ij}(\langle o, \Box, op \rangle)$. If $\beta = 0$, $\tau_{ij}(\langle o, \Box, op \rangle) := \sigma_{ik}(\langle o, \Box, op \rangle \cdot \tau_{ij}(\langle o, \Box, op \rangle)$. If $\beta = 1$, τ_{ij} is not changed.

3.3 Ranking factors

The trustworthiness $\tau_{ij}(\langle o, \Box, op \rangle)$ shows how much a peer p_i trusts another peer p_j with request to an access request $\langle o, \Box, op \rangle$. Another point is how much a peer p_i is trusted. If a peer p_i is trusted by the more number

Table 1. Satisfiability σ_{ij}

request q	state of p_i	state of p_j	$\sigma_{ij}(q)$
$\langle o, \mid, \_\rangle$	$p_i \nmid o$	$p_j \mid o$ $p_j \rightarrow (p_k \mid o)$ $p_j \rightarrow (p_l \rightarrow^* (p_k \mid o))$ others	1 δ $\delta \cdot \sigma_{lk}(\langle o, \mid, \_\rangle)$ 0
$\langle o, \models, op\rangle$	$p_i \models_{op} o$ $p_i \nmid o$	$p_j \mid o$ $p_j \rightarrow (p_k \mid o)$ $p_j \rightarrow (p_l \rightarrow^* (p_k \mid o))$ others	1 δ $\delta \cdot \sigma_{lk}(\langle o, \mid, \_\rangle)$ 0
$\langle o, \models, op\rangle$	$p_i \not\models_{op} o$ $p_i \mid o$	$p_j \models_{op} o$ and $p \mid o$ $p_j \models_{op} o$ $p_j \rightarrow (p_k \models_{op} o)$ $p_j \rightarrow (p_l \rightarrow^* (p_k \models_{op} o))$ others	1 δ δ^2 $\delta^2 \cdot \sigma_{lk}(\langle o, \models, op\rangle)$ 0
$\langle o, \models, op\rangle$	$p_i \not\models_{op} o$ $p_i \mid o$	$p_j \models_{op} o$ and $p_j \vdash_{op} o$ $p_j \rightarrow (p_k \models_{op} o)$ and $p_j \rightarrow (p_k \models_{op} o)$ $p_j \rightarrow (p_l \rightarrow^* (p_k \models_{op} o))$ and $p_j \rightarrow (p_l \rightarrow^* (p_k \vdash_{op} o)$ others	δ δ^2 $\delta^2 \cdot \sigma_{lk}$ ($\langle o, \models, op, \rangle$) 0
$\langle o, \vdash, op\rangle$	$p_i \nvdash_{op} o$	$p_j \vdash_{op} o$ $p_j \rightarrow (p_k \vdash_{op} o)$ $p_j \rightarrow (p_l \rightarrow^* (p_k \vdash_{op} o))$ others	1 δ $\delta \cdot \sigma_{lk}(\langle o, \vdash, op\rangle)$ 0

$0 < \delta \leq 1$

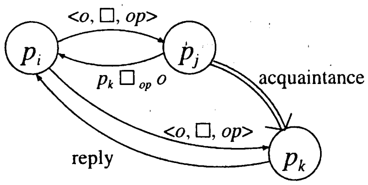

Figure 6. Acquaintance.

of peers, p_i is considered to be more trustworthy. The *rankingfactor* $\rho_i(\langle o, \square, op\rangle)$ shows how much a peer p_i is trusted by other peers with respect to an access request $\langle o, \square, op\rangle$ for an object o, a P2O relation $\square$, and a method *op*. In this paper, a peer p_i is considered to be *more trusted* than another peer p_j if more number of peers trust p_i than p_j. The ranking factor $\rho_i(\langle o, \square, op\rangle)$ is defined as follows:

$$\rho_i(\langle o, \square, op\rangle) := \sum_{p_j \in view(p_i)} \tau_{ji}(\langle o, \square, op\rangle)/|view(p_i)|. \quad (1)$$

Suppose there are four peers p_1, p_2, p_3, and p_4 where $view(p_1) = \{p_2, p_3, p_4\}$. Suppose $\tau_{21}(\langle o, \models, op\rangle) = 0.8$, $\tau_{31}(\langle o, \models, op\rangle) = 0.5$, and $\tau_{41}(\langle o, \models, op\rangle) = 0.6$. Here, the ranking factor $\rho_1(\langle o, \models, op\rangle)$ of the peer p_1 is (0.8 + 0.5 + 0.6) / 3 = 0.63. Each peer distributes the satisfiability and ranking factor to the acquaintance peers. Each peer p_i calculates the ranking factor p_i from the satisfiability information sent by the acquaintances.

4 Charge-based Flooding (CBF) Algorithm

An application first sends an access request $\langle o, \square, op\rangle$ to a peer named *initial peer*. Then, the access request is forwarded to other peers if objects satisfying the requirement are not obtained in the peer. A peer which receives an access request is a *current* peer. Here, suppose there are multiple peers to which the access request can be sent to find the target peers. Even if there might be bigger possibility to find a solution in one route, a same integer value of TTL or HTL is assigned for an access request on every route in traditional flooding algorithms. We newly introduce a concept of *charge* which is given to an access request and which shows the total amount of allowable communication overheads, i.e. number of messages to be transmitted. The more an access request is charged, the more number of peers can be accessed.

1. First, a surrogate which is granted an access right $[o, op]$ is found. If found, the application negotiates with the surrogate.
2. If any surrogate is not found or no surrogate agrees on manipulating the object o, a granter peer of o is searched. If a granter peer is found, the application negotiates with the granter peer to grant $[o, op]$.

First, a peer tries to find surrogates of an object o in the acquaintances. If found, the peer asks the surrogate to manipulate a target object on behalf of the peer. If not found, the peer looks for granters of the object o to obtain access rights on the target object. The peer negotiates with the granter peer to obtain access rights on the *target object*. If obtained, the peer manipulates the target object by itself.

An access request A is charged with some integer value V named *charge*, *A.charge* := V. The access request A is sent to an acquaintance peer p. Here, *A.charge* is decremented by one, *A.charge* := *A.charge* - 1. If A is not satisfiable on manipulating objects in the peer p and is still charged, a set *Cand(A, p)* of candidate acquaintances of the peer p is found. An access request A is *hopeful* on a peer p if *Cand(A, p)* $\neq \phi$. Otherwise, an access request A is *hopeless*. For the hopeful access request A, some acquaintances *Target(A, p)* ($\subseteq$ *Cand(A, p)*) are selected. If $|$*Target(A, p)*$|$ > 1, i.e. *Target(A, p)* = $\{p_1, \ldots, p_m\}$ ($m \geq 1$), A is split into access subrequests $A_1, \ldots, A_m$ where each access subrequest A_i is sent to a peer p_i ($i = 1, \ldots, m$). Here, the charge is allocated to the access subrequests $A_1, \ldots, A_m$ based on the trustworthiness and ranking factor. Let τ_i be the trustworthiness factor of an acquaintance peer p_i for a peer p. Let ρ_j show a ranking factor of a peer p_j.

- If p's acquaintance peers know something about an object o, A_i*.charge* := *A.charge* $\cdot \gamma_i$ where $\gamma_i = \rho_i \cdot \tau_i \,/\, \sum_{j=1}^{m} \rho_j \cdot \tau_i$.
- Otherwise, A_i*.charge* := *A.charge* $\cdot \gamma_i$ where $\gamma_i = (\rho_i \,/\, \sum_{j=1}^{m} \rho_j)$.

That is, the more trustworthy and more trusted a peer p_i is, the larger amount of charge is allocated to an access subrequest A_i to the peer p_i.

Suppose objects are manipulated in a peer p for an access request A. The access request A carries a variable *A.state* whose initial value is U (unsuccessful). If an access request A is performed on the peer p_i, *A.state* is charged with S (successful). After the object o is manipulated by an access request A, *A.state* := S. "*A.state* = S" means that the access request A has so far visited some peer where objects are successfully manipulated.

If *A.charge* = 0, an access request A cannot be anymore forwarded to other peers. The response of the access request A returns to the preceding peer from the current peer p if *A.state* = S. Otherwise, A is discarded. In another case, A is *hopeless*, i.e. *Cand(A, p)* = ϕ but *A.charge* > 0. The response of A returns to the preceding peer p'. The access request A waits for responses from the other access subrequests. If the response of another access subrequest A' returns to the peer p', *A.charge* := *A.charge* + A'*.charge*. *A.state* := S if *A.state* = U and A'*.state* = S. Suppose the responses of all the access subrequests return to p'. If *A.charge* = 0, the response further returns to the preceding peer. *Target(A, p')* := *Cand(A, p')* - *Target(A, p')*. If *Target(A, p')* $\neq \phi$, A is issued to peers in *Target(A, p')*.

5 Conclusion

We discussed how to detect and manipulate objects in a P2P overlay network. We discussed how a peer can access and manipulate objects distributed in peers and can grant access rights to other peers. Types of acquaintance relations are newly discussed with respect to types of service of each peer, i.e. what objects each peer holds can manipulate, and can grant access rights. Based on the acquaintance relations, we defined the satisfiability of an access request in terms of types of service. Then, we defined the trustworthiness of each acquaintance and the ranking factor of each peer based on the satisfiability. Then, we discussed the charge-based flooding (CBF) algorithm through cooperation of acquaintances. Here, the more trustworthy area is more deeply searched in the P2P overlay network.

References

[1] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proc. of the Workshop on Design Issues in Anonymity and Unobservability*, pages 311–320, 2000.

[2] A. Crespo and H. Garcia-Molina. Routing Indices for Peer-to-Peer Systems. In *Proc. of the 22nd IEEE ICDCS*, pages 23–32, 2002.

[3] F. D. Ferraiolo, D. R. Kuhn, and R. Chandramouli. *Role-Based Access Control*. Artech House Publishers, 2003.

[4] Y. Liu, Z. Zhuang, X. Li, and M. N. Lionel. A Distributed Approach to Solving Overlay Mismatching Problem. In *Proc. of the 24th IEEE ICDCS*, pages 132–139, 2004.

[5] Napster. http://www.napster.com.

[6] Oracle Corporation. *Oracle8i Concepts Vol.1*, 1999.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A Scalable Content-Addressable Network. In *Proc. of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172, 2001.

[8] M. Ripeanu. Peer-to-Peer Architecture Case Study: Gnutella Network. In *Proc. of International Conference on Peer-to-Peer Computing (P2P2001)*, pages 99–100, 2001.

[9] A. Rowstron and P. Druschel. Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems. In *Proc. of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, 2001.

[10] 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, 2003.

[11] Sybase SQL Server. http://www.sybase.com/.

[12] K. Watanabe, T. Enokido, M. Takizawa, and K. Kim. Charge-based Flooding Algorithm for Detecting Multimedia Objects in Peer-to-Peer Overlay Networks. *Proc. of IEEE 19th Conference on Advanced Information Networking and Applications(AINA-2005)*, 1:165–170, 2005.

[13] K. Watanabe, N. Hayashibara, and M. Takizawa. CBF: Look-up Protocol for Distributed Multimedia Objects in Peer-to-Peer Overlay Networks. *Proc. of IEEE 19th Conference on Advanced Information Networking and Applications(AINA-2005) Journal of Interconnection Networks (JOIN)*, 6(3):323–344, 2005.

[14] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph. Tapestry: An Infrastructure for Fault-resilient Wide-area Location and Routing. Technical Report UCB/CSD-01-1141, University of California, Berkeley, 2001.