

Community Based Discovery in Peer to Peer Networks

Akihiro Enomoto

Guidance

**Professor
Associate Professor**

**Masao FUKUSHIMA
Tetsuya TAKINE**

2000 Graduate Course

in

Department of Applied Mathematics and Physics

Kyoto University

February 2002

Abstract

Existing discovery algorithms for peer to peer networks are based on broadcast of query messages over the relationship connectivity among entities in the network. A query message contains search key words, and it is broadcast with a TTL (time to live) value to limit the scope of search and to reduce high overhead associated with broadcast. Although the use of TTL reduces the amount of overhead, it is difficult to determine an optimal TTL value for a given network, and thus, scalability of existing discovery algorithms may be limited. In addition, in existing discovery algorithms, all search target entities that match the query keywords return a query hit, independent of if they are considered useful in the past by users (i.e., query originator), potentially leading to a large number of non-useful query hits returning to a user. In this paper, we propose a new discovery algorithm for peer to peer networks based on the community and user preference. In the proposed discovery algorithm, each entity gradually acquires keywords of entities that it has a relationship with and represents a community (i.e., a group of entities that it has a relationship with). In addition, users in the proposed discovery algorithm express their levels of satisfaction with the query hits they receive. In the proposed discovery algorithm, these communities and user satisfaction levels are used in determining where a query message may be forwarded to in search and whether an entity received a query message should return a query hit to the query originator. Community based query message forwarding reduces unnecessary forwarding of query messages, makes the discovery process robust to dynamic changes in the entity relationship connectivity, and eliminates the need for the use of TTL. Query hit returns based on user satisfaction levels reduce the number of unnecessary hits returns.

In this paper, we examine characteristics of the proposed discovery algorithm and obtain various performance measures through simulations. Our simulation results show that the proposed algorithm is scalable and robust to dynamic changes in a network.

1	Introduction	4
1.1	Peer to Peer Networks.....	4
1.2	Existing Discovery Schemes in Peer to Peer Networks.....	4
2	Assumptions and Terminology	5
2.1	Cyber entity	5
2.2	Relationship	6
2.3	Cyber entity Attributes	6
3	Proposed Discovery Algorithm Scheme	9
3.1	Community.....	9
3.2	Discovery Algorithm – Overview –	10
3.3	Discovery Algorithm – <i>Forwarding of a query packet</i> –	11
3.4	Discovery Algorithm – <i>Query Hit Backwarding Part</i> –	17
3.5	Discovery Algorithm – <i>Reward Feedback Part</i> --	19
4	Numerical Results	21
4.1	Simulation Assumptions	21
4.2	Simulation 1– Simple Probabilistic Forwarding –	23
4.3	Simulation 2 Discovery on a Static Network.....	24
4.4	Simulation 3 Discovery on a Dynamic Network.....	28
5	Conclusions and future work	33

1 Introduction

1.1 Peer to Peer Networks

Recently Peer to Peer Systems such as Gnutella[1], Freenet[2] and NeuroGrid[3], etc are becoming commonplace as a means for users connected on large networks to take advantage of the vast resources available to them. A Peer to Peer system is different from the traditional centralized client/server model because the applications involved act as both clients and servers. That is to say, while they are able to request information from other servers, they also have the ability to act as a server and respond to requests for information from other clients at the same time. In other words, we can say that no central or coordinating entities exist in Peer to Peer networks.

By the way, Internet is growing at an exponential rate and has changing our life since the dawn of its history[4]. With the Internet, people can communicate with others around the world without any delays and also buy items from all over the world, while staying at home. However, there are some indications of current Internet system declining. The Internet in the past has been designed as a static network. With the recent widespread of mobile phones, and their email and web applications, the Internet now is becoming highly dynamic supporting mobile users. The Network is on the point of being dynamic. We are in urgent necessity to build a new network paradigm in a different approach to be adapting such a dynamic network[5][6][7]. Now for such a demand we put our hope on a Peer to Peer system

Since data and applications migrate around the network to optimize resource usage and response time, a network must provide a mechanism to locate such mobile network objects (data, users, and applications)[8]. One potential method of searching for an object located in a network is to have each network object contain links to, and information about, several other network objects. A search may use such links to traverse a network and information to guide the search more quickly towards the desired network objects. In this paper, we develop and investigate a new discovery technique that provides such functionality.

1.2 Existing Discovery Schemes in Peer to Peer Networks

Compared with a centralized server system, it is harder to discover a desired entity or a desired type of entities in a Peer to Peer network because a peer does not have access to the global status of the network, and thus, is not able to locate where the desired entity is.

In most of existing discovery schemes for peer to peer networks, therefore, rely on broadcasting of a query message. Although the broadcast based discovery is simple, this causes some serious problems[9].

One major problem is that the broadcast potentially waste too much network resources. To reduce the overhead due to broadcasting of query messages, most Peer to Peer discovery schemes use a TTL value to limit the scope of discovery or the scope of query broadcast. The TTL value in the query message specifies how many times the query message may be forwarded by peers before it is removed from the network. Although the use of TTL values in query messages reduces the overhead due to broadcast, it introduces a new problem, namely, it is far from obvious to determine an appropriate value for TTL. Most Peer to Peer applications (including peer to peer discovery schemes) set the TTL value to 7, the value empirically arrived at. The choice of this TTL value may or may not be right, especially given that a current E-mail server (a SMTP server) takes sometimes 20-hops to deliver to a target SMTP Server.

Another major problem with the existing broadcast based discovery schemes is that potentially a large number of query hits may return to the query originator. In existing discovery schemes, searches are based on keywords, and entities that match the keywords contained in a query message return a query hit[10]. With the broadcast based query forwarding, all entities that match keywords return a query hit. The query originator may be flooded with hits that contain satisfactory information and also with hits that may not contains satisfactory information.

2 Assumptions and Terminology

In the following, we describe assumptions and notations that we use in this paper to describe the proposed discovery scheme.

2.1 Cyber entity

In the proposed discovery mechanism, we assume that the discovery mechanism abstracts data, information, applications and users as network objects (referred to as entities in the rest of the paper), locatable on the network[11]. Each network object (cyber entity) is created with a globally unique identifier and is capable of moving from one network node to another. Each network object (cyber entity) also contains descriptive information (keywords) about itself, such as information describing whether it is data or an application, its type of service, or additional keywords describing the service. Keywords

allow searches to distinguish between applications of the same type.

2.2 Relationship

In the proposed discovery mechanism, each cyber entity has a list of relationships to other entities. A cyber entity has a relationship with another cyber entity when it knows about the other cyber entity (i.e., when it knows the identifier, address and location of the other cyber entity). A relationship may also contain descriptive information about the entity that it knows.

A cyber entity may establish a relationship with another cyber entity through actively searching for other cyber entities nearby, through introduction via other entities, and through discovery-related interactions. Actively searching for other entities involves broadcasting an inquiry message using the network's node level connectivity to learn about nearby entities. (Once a nearby entity is found, a relationship can be established.) A cyber entity can introduce two entities that it has relationships with to each other by having one cyber entity pass knowledge about itself to the other entity. Discovery related interactions occur when entities communicate during the discovery process, and can be used to introduce other cyber entities. Through these mechanisms, a set of relationships is built and adjusted to form a network of relationship over all entities.

Using relationships among entities provides an adaptive and decentralized mechanism for searches[12]. Searches originate from an entity and travel from entity to entity based upon relationships. Even when cyber entities migrate, relationships are maintained, as entities are aware of which other entities they have relationships with, and the migrating entity update the other entity's relationships when it changes location on the network. This provides a robust environment for the movement of entities. Since each cyber entity only uses its own relationships to perform searches, the search method based on relationships is entirely decentralized[13].

2.3 Cyber entity Attributes

In this paper, we consider a peer to peer network that consists of cyber entities connected through relationships. A cyber entity is an abstract of various system resources such as information (e.g., a web page) and a node. Relationship represents a pointer established

between two cyber entities, and cyber entities communicate with each other via its relationship. Namely, in our discovery mechanism, query messages are forwarded from a peer to a peer using the relationship connectivity.

Attributes of a cyber entity include an ID (CE ID), an address (CE address), a set of (one or more) keywords. CE ID is given to a cyber entity when it is created, and it does not change during the lifetime of a cyber entity. CE address refers to the location of a cyber entity (e.g., an IP address of the node where the entity currently resides), and it changes everytime a cyber entity migrates from a node to a node.

Cyber entity attributes also include a set of keywords for the cyber entity. It describes characteristics of neighboring cyber entities that the given cyber entity knows about (including the given cyber entity itself). Associated with each keyword is the strength and a CEID of a neighboring cyber entity that resulted in addition of the keyword to the keyword set. Keyword strength of a cyber entity is a measure of how useful the cyber entity has been in the past in discovering cyber entities containing the given keyword[14].

Note that the set of keywords is dynamic. For instance, a keyword (or keywords) is added (with an initial strength value) to the keyword set when a cyber entity receives a successful query hit (from a neighboring cyber entity) for a new keyword (new keywords) that it has not seen. A keyword is removed when its keyword strength becomes zero.

The following **Table 1** shows what information is kept in cyber entity relationship table. A cyber entity can communicate the other CEs on this table.

<i>Attribute</i>	<i>Explanation</i>
<i>CE ID</i>	cyber entity ID of itself (cyber entity A)
<i>CE address</i>	reference to the location of itself (cyber entity A)
<i>Keyword 1</i> CE ID Keyword 1 Strength	a keyword CE ID of a cyber entity (say, cyber entity A itself) who resulted in addition of keyword 1 usefulness of cyber entity A in discovering cyber entities with keyword 1
<i>Keyword 2</i> CE ID Keyword 2 Strength	a keyword CE ID of a cyber entity (say, cyber entity A itself) who resulted in addition of keyword 2 usefulness of cyber entity A in discovering cyber entities with keyword 2
<i>Keyword 1</i> CE ID Keyword 1 Strength	a keyword CE ID of a cyber entity (say, cyber entity B) who resulted in addition of keyword 1 usefulness of cyber entity A in discovering cyber entities with keyword 1
<i>Keyword 2</i> CE ID Keyword 2 Strength	a keyword CE ID of a cyber entity (say, cyber entity C) who resulted in addition of keyword 2 usefulness of cyber entity A in discovering cyber entities with keyword 2
<i>Keyword 3</i> CE ID Keyword 3 Strength	a keyword CE ID of a cyber entity (say, cyber entity C) who resulted in addition of keyword 3 usefulness of cyber entity A in discovering cyber entities with keyword 3

Table 1 Relationship table of cyber entity A

In the following description of the proposed discovery scheme, we assume that there are no restrictions on the number of keywords that a cyber-entity keeps in its relationship table.

3 Proposed Discovery Algorithm Scheme

In this section, we explain our proposed community based discovery algorithm scheme in peer to peer networks.

3.1 Community

3.1.1 Desired Peer to Peer Discovery

Existing discovery algorithms for peer to peer networks are based on broadcast of query messages containing search keywords with a use of TTL (time to live) to limit the scope of search to reduce high overhead associated with broadcast. Although the use of TTL reduces the amount of overhead, it is difficult to determine an optimal TTL value for a given network, and thus, scalability may be limited. In addition, in existing discovery algorithms, all search target entities that match the query keywords return a query hit, independent of if they are considered useful in the past by users (i.e., query issuerers), potentially leading to a large number of non-useful query hits returning to a user.

3.1.2 Concept of Community in the Proposed Discovery Algorithm

In the following, we illustrate the concept of community used in our discovery scheme.

To illustrate, consider the following scenario. 'Person A asked person B for information X. B does not have information on X, and thus, B in turn asked Person C. C has information on X and passed the information to B; B in turn passed the information to A.'

In this simple scenario, we observe the following points.

(1) Although B simply passed the requested information X from C, it looks to A as if B itself has the information X B appears to A as an expert on information X (i.e., someone who has information on X).

Observation (1) implies that, in discovery, it is not necessary to specify the actual location of information being sought; rather, it is sufficient to know the direction of information (or who is connected to the information being sought). In other words, if, in the previous example, A remembers B as information X (i.e., an associates B with keyword X in A's relationship table), it helps future discovery process; Note that B is not aware that A remembers B as information X.

(2) B does not know that A thinks of B as a specialist on information X

Observation (2) implies that relaying of information (queries and query hits) preserves anonymity of query originators and those who return query hits. (In the previous example, it may appear that B is aware of who the query originator is and who returned the query hit. This is not true, as A might be simply passing a query to B from its neighbor, and C might be passing B a query hit from its neighbor.)

(3) Although A received information X from C (through B), identity of C is hidden from A. Anonymity is preserved between the query originator and the person who returned the query hit.

(4) If A would like to find information X next time, A would probably ask B again as A now knows that B was successful in finding information X in the past. Note, however, that relying on the past history (of B successfully passing the information X to A) may not always lead to a successful search as the relationship between B and C might change, or there may be another person who may find information X quicker than B

To summarize, a relationship is not only a pointer to locate a relationship partner, but also a pointer that includes information (keywords) on its partner. The relationship information is not necessary to describe the relationship partner, but may also describe peers that the relationship partner knows about. This concept of a cyber entity representing peers that it knows about is referred to as community.

3.2 Discovery Algorithm – Overview –

The proposed discovery algorithm consists of the following five phases.

1. Creation of a query message

A query originator cyber entity creates a query message. A query message contains the message ID (globally unique ID), message type (indicating that the message is a query), a set of keywords to specify the attributes of the cyber entity (cyber entities) being sought.

2. Forwarding of a query message

In deciding to which cyber entities to forward a query message to, a cyber entity chooses, according to a given forwarding policy, a cyber entity (or cyber entities) from among the cyber entities that it has a relationship with, and forwards the query to the chosen cyber entities. The forwarding policy used in the proposed discovery

mechanism is a probabilistic forwarding based on the strength of a keyword. (See a later section for details of the forwarding policy used in the proposed discovery mechanism.)

3. Creation of a query hit

When a cyber entity receives a query message, it first examines if it satisfies the search, and if it does, it responds to the search. In the proposed discovery mechanism, if a cyber entity satisfies the search or not is probabilistically determined based on the keyword strength. When the cyber entity itself does not satisfy the search, it uses its relationships to forward the query packet to other cyber entities. (See the phase 2 above for the query forwarding phase.)

4. Returning of a query hit

Query hit backtracks the path that the query message traversed and returns to the search originator.

5. Forwarding of a reward message

Upon receiving a query hit (or query hits), the search originator examines the hit (or each hit), and determines how satisfactory the query hit is, and based on the degree of satisfaction, the search originator probabilistically creates a reward message and sends the reward message along the path that the query hit returned. As the reward message propagates over the path, it increases the keyword strength and adds a new keyword to intermediate cyber entities, if they have not see the keyword. Reward is forwarded along the path that a query hit was returned on to increase some keyword strengths of the partners. This reward makes a community.

3.3 Discovery Algorithm – *Forwarding of a query packet* –

As described earlier, a query packet contains parameters (i.e., a key word) describing the target(s) of query. Upon receiving a query packet, a cyber entity first examines if it has seen the query in the past. If it has seen the query message before, then, the query message is discarded. If it has not, then, the cyber entity examines if it satisfies the query or not. If it satisfies the query, then, it uses its relationships to forward the search request to other network objects. Searching is improved by using the additional relationship information (i.e., descriptive information about other cyber entities such as keywords neighboring cyber entities know about and their keyword strength (i.e., a

measure of how useful neighboring cyber entities have been in the past in finding such keywords). Based on the parameters of the search and the descriptive information contained in a relationship, a decision can be made regarding which relationship is more likely to lead towards a cyber entity (cyber entities) that meets (meet) the search parameters and a search request may be forwarded to such cyber entities. Details of the query forwarding process in the proposed discovery mechanism are described below.

3.3.1 Check Creation of a Message Forwarding List

Upon receiving a query packet, a cyber entity first examines if it has seen the query in the past by checking the message ID of the query packet. Note that each query message contains the unique ID, and thus, by examining the message ID, a cyber entity can tell if it has seen the same query before or not. If it has not, it then creates a new entry in the query message forwarding list.

<i>Message Forwarding List</i>		
<i>Received Message. ID</i>	<i>Received From</i>	<i>First Arrival Time</i>
Msg ID [3280]	CE-A , CE-B	15.16

Table 2 a Message Forwarding List

Note that a query packet in our discovery may be forwarded to multiple paths (as we will see later in this section), and thus, copies of the same query packet may be received by a cyber entity. The “received message ID” field in the message forwarding list records the message ID of the newly received query packet. The “received from” field records the ID of the cyber entity that forwarded the query message and the “arrival time” field records the arrival time of the query. After completing the message forwarding list for the newly received query packet, the cyber entity then decides which neighboring cyber entities to forward a query packet to.

If the cyber entity has already seen the received query packet, then, it updates the “received from” field by adding the cyber entity that forwarded the query packet. (For instance, if the same query packet is forwarded by two cyber entites, A and B, then, the “received from” field records both A and B, as shown in Table.) The “Received message ID” field stays the same. The “arrival time” field does not change so that this field contains the arrival time of the first query packet among the query

packets with the same message ID number.

The “arrival time” field is used to determine when to discard an entry in the message forwarding list to avoid infinite growth of the list. The “received from” field is used to determine to which cyber entities a query hit will be forwarded to in the query hit backwaring process (to be explained in section.)

3.3.2 Forwarding of a Query Message

When a cyber entity knows it has never received the message in the past, the cyber entity will decide some partners to forward (including returning a query hit) by the calculation of forwarding priority.

As mentioned earlier, it is not always a good policy to ask only for the peer which often returns good answers in the dynamic network. We consider forwarding criteria needs factor of broadcast to be adaptive to dynamic network. Our proposed forwarding method must be required the following:

1. Give forwarding priority to the peer knows about desired information
(From Broadcast to Multicast)
2. Leave the probability to forward for the peer doesn't know desired information
(From Multicast to Broadcast)

To satisfy above two conditions, we proposed probabilistic forwarding.

In this paper, we proposed two forwarding priority model from different viewpoint using concept of community. One model is based on getting best route when there are many routes (Like Unicast). The other model is based on getting all the better routes when there are many routes. (Like Multicast) In this paper, we provide two forwarding priority models, Rate and Sigmoid, as examples above model.

Before explaining our two probabilistic forwarding, we explain broadcast forwarding.

3.3.2.1 Broadcast

We assume that a peer has n-relationship partners and receives a message. The peer will forward the incoming query message to all of its directly connected peers, except the one that delivered the incoming query unless TTL in the message reaches 0. TTL is the number of times the message will be forwarded by

peers before it is removed from the network. Each peer will decrement the TTL before passing it on to another peer. When the TTL reaches 0, the message will no longer be forwarded. This mechanism avoids the flooding of message in the network.

3.3.2.2 Probabilistic Forwarding – Rate –

The forwarding policy of Rate is to give the more priority for the partner has the higher keyword strength in the relationship partner. We can get selection probability for partners by the Rate.

$$\text{Selection Probability of partners } n = \frac{x_n}{\sum_{\{\text{partner and itself}\}} x_k} \dots \dots \dots (1)$$

where x is keyword strength.

Initial Keyword strength means chance of success keyword strength when unsetting the keyword on relationship.

3.3.2.3 Probabilistic Forwarding – Sigmoid –

This probabilistic forwarding is based on the concept to calculate selection probability only from the partner's keyword strength taking no thought of others. As an example such a calculation, we show Sigmoid using sigmoid function. The reason we choose this function as an example is that the sigmoid function returns the value from 0 to 1 any x., so we can use the value as probability

$$\text{Selection Probability} = \frac{1}{1 + \exp\left\{-\left(\frac{x}{x_0} - \alpha\right)\right\}} \dots \dots \dots (2)$$

x is a value of keyword strength that corresponds to the keyword in a query.
 x₀ is constant. and $\alpha = \log(m - 1)$ when $m > 1$ m is integer

3.3.2.4 Discovery without TTL

These probabilistic forwardings may let us release the restriction of TTL in a discovery. Current peer to peer systems use broadcast forwarding, so there is a necessity to use TTL by such a forwarding policy in order to avoid explosive query increase. In consequence, TTL restricts discovery range of the network, so that it

means it is impossible to discovery in the larger space. We considered a discovery should ideally be following for anyone:

(1) It doesn't matter how many hop counts a query takes to discover when the query reaches the best solutions.

(2) A query should be rejected when the query doesn't reach any best solutions.

However, TTL mechanism doesn't satisfy above both of these at all.

We introduce probabilistic forwarding as example of community based forwarding method. Probabilistic forwarding might prevent a discovery from explosive query increase.

A point to notice on the probabilistic forwarding is the average forwarding number. When the average forwarding number is always more than one, it may sometimes cause a flooding of queries in the network.

Two probabilistic forwarding we bring up are based on two different criteria, we mentioned earlier in this section. Therefore, there is following difference between two our instance probabilistic forwarding about the expected number of forwarding.

By Rate, the number is always 1 (As you know later, the number is always less than 1. We explain it later.)

· By Sigmoid, the number is not constant. it is 1 or more than 1.

(As you also know later, the initial number is always less than 1.)

In this paper, we didn't optimize these probabilistic forwarding.

3.3.2.5 Forwarding probability toward the peer that delivered the incoming query

In this paper, we choose the forwarding calculation policy to remove a probability toward the Cyber Entity that delivered the incoming query (See Fig. 1, CE-A) Fig. 1 shows that the difference between the forwarding probabilities including the

probability toward the cyber entity that delivered the incoming or not. This problem is not just simple one. However, we choose removal policy in order to avoid complicated problems accomplished without removal policy.

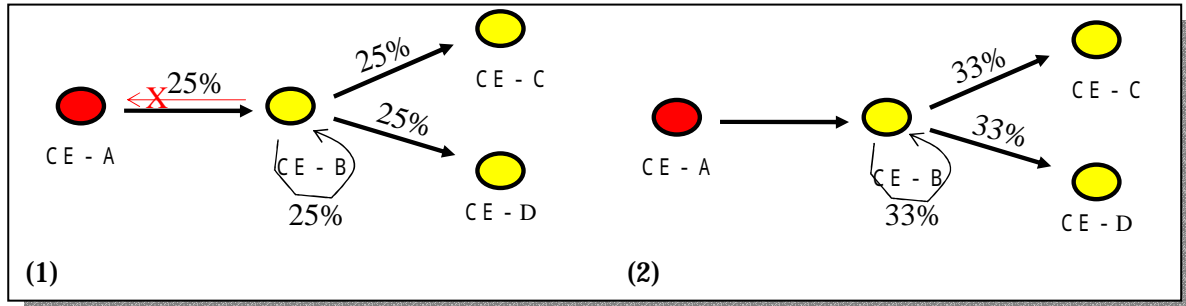


Fig. 1 Forwarding probability toward the peer that delivered the incoming query

3.3.3 After the calculation of the selection probability

After calculation of each partner's selection probability to forward a query, the Cyber Entity decides to forward probabilistically. In this paper, a cyber entity decides to forward on every partner and does not use roulette selection.

3.3.4 Adjusting keyword strength

We take the following policy to adjust keyword strength.

- After forwarding to a partner, decrease the partner's keyword strength
- When receiving a reward, increase the partner's keyword strength.

(We will explain later.)

If a cyber entity doesn't set the query's keyword on the partner, the cyber entity doesn't decrease any keyword strength on the partner. The reason is that the cyber entity forwards a query to the partner with at its discretion. There is no blame for the partner. And the cyber entity will get a good answer from the partner next time though it couldn't get good one last time. Because the network is dynamic, so no Cyber Entity can guess what the result will be.

Negative Keyword Strength

Suppose a Cyber Entity can set negative keyword strength on its partners. A cyber entity is allowed to set negative keyword strength on the partner that returns no answers. Talking to human world, assuming that you ask something for a certain person, however the person returns no answer over and over, In this case, you'll be unwilling to ask for the person because you memorise that the person returns nothing. If there is no negative keyword strength mechanism, a cyber entity won't do like that.

Therefore, it seems to be a good policy setting negative keyword strength. However, this problem has dynamic network and memory capacity problems.

First, dynamic network is unpredictable. There is always some probability of good information incoming or its birth in dynamic networks. So a Cyber Entity shouldn't fully rely on historical data.

Second, suppose that every time a cyber entity forwards a query to some partners and there is no response from those, and the Cyber Entity sets some negative keyword strength on those partners. However, we can expect there are a lot of memories of the Cyber Entity to store those negative strengths. It is waste to store negative keyword strength such meaningless information. So we choose the policy that Keyword Strength won't be under the certain value, initial keyword strength.

3.4 Discovery Algorithm – ***Query Hit Backwarding***

CEs meeting minimum query conditions probabilistically return a query hit along the path that a query was forwarded on.

3.4.1 Returning a query hit policy in peer to peer networks

Discovery of current peer to peer systems is based on keyword matching. Search accuracy falls far short to recent centralized search engines. In addition, the problem we should take care in design is information sunken problem. Information sunken problem is following cases:

- (1) A client cannot pick one of best solutions out of a great number of answers.
- (2) A client will choose the answers coming earlier, so the client cannot get best solutions.

As we mentioned earlier, although centralized server systems aim at linking between the client who is looking for information and the provider that has best information for client, in peer to peer networks, it is important for a peer not only to answer the query, but also to leave it to its partner.

We see that it may be good for systems that a cyber entity doesn't always return a query hit everytime even if the cyber entity matches a query keyword. However, what benefit does the cyber entity get without returning a query hit?

If a cyber entity continues to provide poor results, the cyber entity would get to be regarded as worthless by others and might never receive a query by others.

To come up to this, we set a cyber entity to have keyword strength, and we make a cyber entity calculate forwarding probability to decide returning a query hit as well as calculate forwarding probability toward individual partner. The concept of community let us think of forwarding as returning a query hit.

3.4.2 Returning a query hit when no keyword matching

When calculating forwarding probability toward probability toward each partner, we mentioned earlier, a cyber entity uses prospective probability if not setting keyword strength on its relationship. Similarly, a cyber entity is allowed to return a query hit probabilistically even if the cyber entity matches no keyword in the query. One of shortcomings in keyword matching is that a peer must have the same keyword as query's. This mechanism would break the shortcoming. We explain about this later.

3.4.3 Routing of query hits

Storing a message flow for a short while, a cyber entity can return a query hit along the path that a query was forwarded on. Then, a cyber entity can behave as a multicast by its message forwarding list.

Fig. 5 Tandem Connection Network. illustrates an example of query hit routing.

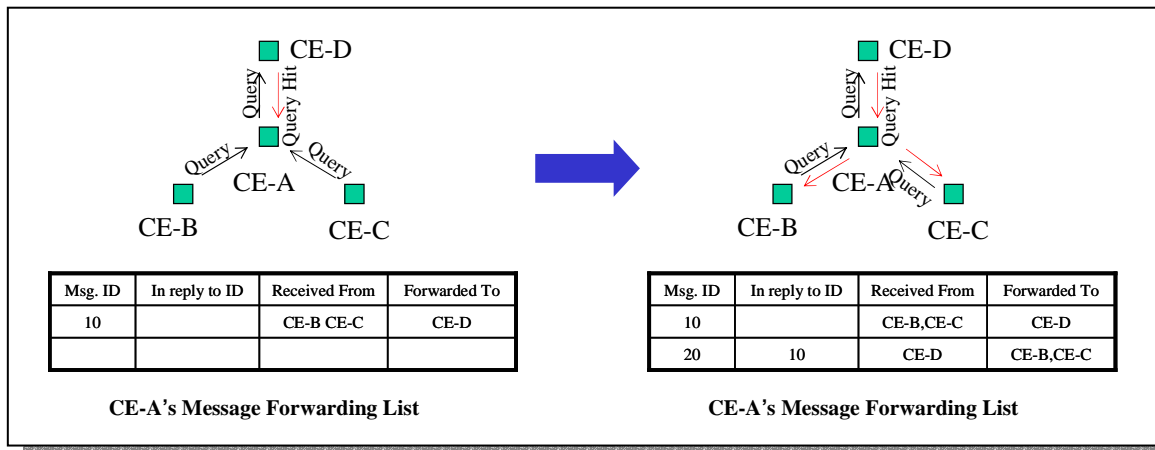


Fig. 2 Query hit routing

3.5 Discovery Algorithm – *Reward Feedback Part* --

A query originator CE gives a reward only to a CE (CEs) that returned a satisfactory query hit (not to all CEs that returned a query hit). Reward is forwarded along the path that a query hit was returned on to increase some keyword strengths of the partners. A community is created by reward messages.

3.5.1 Reward

A search originator gives a reward for a satisfactory query hit. For a reward is to make up a community, the search originator shall put keyword describing desired community on the reward. The keyword is not restricted to be a query keyword. A search originator can put any keywords on a reward. However, it should pay attention to handle with because the keyword would construct community. Therefore, it had better keep to use general words if it wants to share community. Alternatively, it might be better to put strange words if it does not want to share.

3.5.2 Revising of self CE Description

As mentioned previously, a reward is used to build a community. How it is valid not only for that, but also for revising self CE Description. This idea brought by the concept of community would give scalability on peer to peer. We illustrate this by the following simple example:

	Original Self CE Description:	a,
	(Incoming Reward Keywords:	<u>b</u>)
→	New Self CE Description:	a, <u>b</u>

3.5.3 Routing of a Reward message

As well as query hit routing, a reward is also forwarded along the path that a query hit was returned on.

3.5.4 Adjusting Keyword Strength – Building a community -

To construct a community, a cyber entity increases some keyword strength of the partners to be rewarded on their relationships. If its relationship does not contain keyword strength to increase, the cyber entity adds the keyword in the reward on its relationship.

4 Numerical Results

In this section, we evaluate various characteristics of the proposed discovery mechanism through simulations. We first explain the assumptions made in our simulations and the configurations we simulated.

4.1 Simulation Assumptions

4.1.1 Cyber Entity Migration

In some of the simulation results in this section, cyber entities migrate so that we can examine the efficiency of the proposed discovery algorithm. When a Cyber Entity migrates, it decides the migration angle $(0 \leq \theta < 2\pi)$, as well as the migration distance, d . In simulations, the migration angle θ is randomly chosen, and the migration distance is either 3 or 6.

A cyber entity makes decision as to whether it migrates or not according to the Poisson distribution with the rate of 0.8. In other words, at time instances following the Poisson distribution, a cyber entity decides whether it migrates (with probability 0.5) or it stays (with probability 0.5).

4.1.2 Relationship Establishment Policy of Cyber Entities

In our simulations, each cyber entity is associated with a transmission range. A cyber entity can communicate with any and all cyber entities within its transmission range, but it cannot communicate with any cyber entities beyond its transmission range. Thus, in our simulations, a cyber entity establishes and maintains a relationship with each and every cyber entity in its transmission range, but not with cyber entities outside of its transmission range. In other words, in our simulations, a cyber entity, upon receiving a message (i.e., a query packet, a query hit, or a reward packet), it senses its transmission range and updates the relationship table (by removing the cyber entities who are not in the transmission range, and by adding new cyber entities who are in the transmission range). This is necessary as cyber entities in our simulations move to create a dynamic environment.

In the simulation results we present in this section, the transmission ranges of all cyber entities are assumed to be the same, and the radius of the transmission range is assumed to be 35.

4.1.3 Initial Network Topology

In our simulations, we assume 550 cyber entities, each with the same transmission range of 35. We also assume that each and every cyber entity initially has 8 other cyber entities in its transmission range. This results in an initial cyber entity relationship topology of a square topology (See Fig.3). Links of the square topology represents the initial relationship between the cyber entities. Distance between two cyber entities located horizontally or vertically is same and is 23, and diagonal distance between the two cyber entities is in this initial topology.

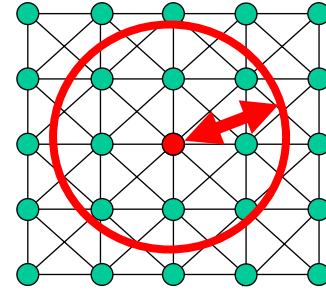


Fig.3 Square Network Topology and transmission radius 35

4.1.4 Keywords and Goodness

In our simulations, we assume a total of 10 types of keywords in the system. Each cyber entity is randomly assigned a single keyword (called a self description keyword) to describe itself.

In reality, a query originator, upon receiving a query hit, examines the hit and determines its level of satisfaction and probabilistically returns a reward packet based on the level of satisfaction. In our simulations, each cyber entity is assigned a goodness value at the beginning of the simulation to indicate the level of satisfaction by the query originator. The goodness value for a cyber entity is randomly chosen between 0 (least satisfied) and 1 (most satisfied), and this goodness value does not change throughout the simulation. In our simulations, upon receiving a query hit from a cyber entity, the query originator examines the goodness value of the query hit, uses the goodness value as the probability to return a reward.

4.1.5 Search Originator Cyber Entity

In reality, any cyber entity can issue a query packet. In our simulations, without loss of generality, we assume that only one cyber entity (with CE ID of 1) issues a query packet. In addition, we also assume without loss of generality, the search originator only issues a query packet containing the same keyword throughout the simulations.

4.1.6 Simulation Cycles

Our simulator is developed using Java 1.3.1 on Windows 2000. One simulation cycle corresponds to the time interval beginning when a cyber entity issued a query packet and ending when all discovery related messages (query packets, query hits, reward packets) resulted from the query disappear from the system. Note that query packets are removed from the system when a cyber entity returns a query hit, or when an intermediate cyber entity probabilistically decides not to forward the query packet to its neighboring cyber entities. Query hit packets are removed at the query originator. Reward packets are removed at the cyber-entity (cyber entities) that returned a query hit. In each of the simulation results that we show in this section, simulator runs for 1,000 cycles.

4.1.7 Message Transmission Delays

In our simulations, we assume that transmission of messages (i.e., query packets, query hits, reward packets) between two cyber takes zero time. This means no cyber entity moves during a discovery cycle.

4.2 Simulation 1 Simple Probabilistic Forwarding

This proposal uses probabilistic forwarding. This simulation shows us how many queries one discovery can forward by simple probabilistic forwarding.

4.2.1 Simulation 1-1– Simple Probabilistic Forwarding in a tandem network –

In this simulation, we arranged cyber entities in a line. (See Fig. 5). The cyber entity on the most left side is a search originator. First, the search originator requests a query and forwards it to next right cyber entities by probability p ($0.1 \leq p \leq 0.99$). Each cyber entity that receives the query forwards it to next cyber entity by the same probability p . We measure how many times a query hops in a discovery by probability p . The results are given in Fig. 6 and Table 3. (Theoretical hops is $p/(1-p)$) From these results, we see that the number of query hops in a discovery is increased exponentially as p is increased. And we see that the number of hops is very low under the probability 0.9. Especially under the probability of 0.5 the number is less than 1 hop. We should take care that probability 50% means a query is forwarded 50 times and dismissed 50 times in 100 discovery.

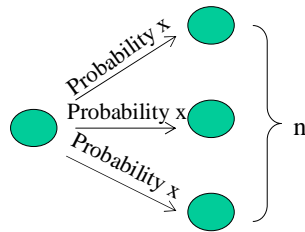


Fig. 4

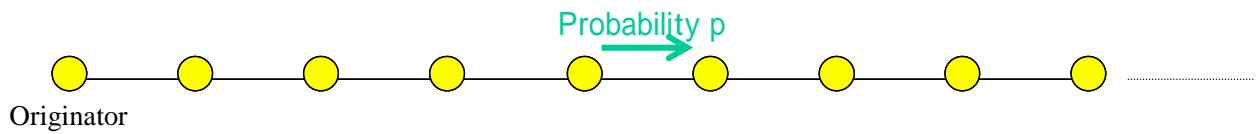


Fig. 5 Tandem Connection Network.

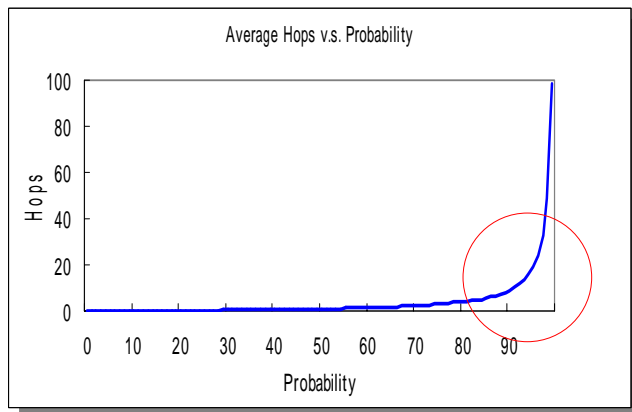


Fig. 6 Average Hops in a tandem network used by various simple probabilities

Probability	Average Hops	Max.Hops	Min. Hops
10%	0.1154	4	0
30%	0.4428	9	0
50%	0.991	14	0
60%	1.5127	17	0
70%	2.3263	25	0
80%	4.0213	41	0
90%	8.8765	85	0
99%	98.4238	787	0

Table 3 Average Hops in a tandem network used by various simple probabilities

4.3 Simulation 2 Discovery on a Static Network

In these simulations, we assume a static network to see how our proposal gives impact to

peer to peer networks. In this static network, no Cyber Entity moves and changes its relationships to others. To see community formation, we measured sum of keyword strength on all relationships at search originator. To estimate discovery range of our proposed scheme, we measured the number of entities required to discover at least one keyword entity. We solved the number of entities required to discover a keyword entity per one keyword entity as discovery efficiency from previous measurement. To estimate overhead on query forwarding, we counted the times of receiving the same query each an entity in a cycle. To confirm validness of our proposed scheme, we measured all goodness of query hit a search originator received and acquired the average.

4.3.1 Community formation

To see community formation, we measured sum of keyword strength on all relationships at search originator. Fig.9 (Rate) and Fig.10 (Sigmoid) show these results. From these results, we can see that the total keyword strength increases throughout the simulation because there is no relationship vanishment in the static network.

4.3.2 Discovery range

To estimate discovery range of our proposed scheme, we measured the number of entities required to discover at least one keyword entity. The results are given in Table 5. The reason that there is the big difference in the number of entities between Rate and Sigmoid is in the difference between unicast (Rate) and multicast (Sigmoid). Another reason is that expected forwarding numbers of Sigmoid is over one in a community. We can see the fluctuation is different from the other from the table. The reason is the higher keyword strength increase, the much more the forwarding tends to unicast in the Rate. In the case of Sigmoid the reason is that the larger a community is, the more expected forwarding numbers is over one.

	<i>Rate</i>	<i>Sigmoid</i>
<i>Average(Cycle 1 – 1000)</i>	5.474	61.900
<i>(Cycle 1-500)</i>	8.744	44.875
<i>(Cycle 501-1000)</i>	3.347	78.822

Table 4 the number of entities receiving a query in one cycle

4.3.3 Discovery efficiency

To estimate discovery efficiency, we divided the number of entities required to discover a keyword entity, we acquired in the previous subsection, by the number of discovered keyword entities. The results are given in Table 6. The former part of Rate (Cycle 1 ~ 500) gives poorer result than broadcast's because a search originator fails to discover any keyword entity in most cycles. In the case of Sigmoid the latter part gives poorer result because the larger a community is, the more entities are discovered.

	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 - 1000)</i>	3.093	4.142	4
<i>(Cycle 1-500)</i>	4.253	4.553	4
<i>(Cycle 501-1000)</i>	2.338	3.819	4

Table 5 Discovery efficiency (the number of entities inquired to discover one keyword entity)

4.3.4 Overhead

To estimate overhead of query duplication, we measured the times of receiving the same query in a cycle. The results are given Table 7. We can see that Rate and Sigmoid cut down the overhead.

	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	1.120	1.772	2.667
<i>(Cycle 1-500)</i>	1.261	1.710	2.667
<i>(Cycle 501-1000)</i>	1.029	1.821	2.667

Table 6 Overhead (the average times of an entity's receiving the same query)

4.3.5 Quality of Query Hits

To confirm validness of our proposed scheme, we measured all goodness of query hit a search originator received and acquired the average. Table 8 shows the results.

The reason that Rate gives poorer results than Sigmoid's is that ATODE!!!!!!

	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	0.918	0.807	0.498
<i>(Cycle 1-500)</i>	0.855	0.832	0.498
<i>(Cycle 501-1000)</i>	0.951	0.789	0.498

Table 7 the average Quality of Query Hits

4.3.6 Summarize

To see comparative performance of our discovery schemes with broadcast, we calculated previous measurements treated broadcast's measurement as 1. The results are given in Table 9. Table 9 also shows the average maximum hops to a keyword entity and maximum hops in a discovery. We can confirm that our proposed discovery schemes give good results except some results about discovery efficiency in a static network.

<i>Static</i>	<i>Discovery Efficiency</i>	<i>Overhead</i>	<i>Quality of Query Hits</i>	<i>Max Hops to solution</i>	<i>Max Hops</i>
Rate	0.773	0.420	1.843	2.032	2.672
(Former)	1.063	0.473	1.715	3.149	3.888
(Latter)	0.584	0.386	1.908	1.310	1.657
Sigmoid	1.035	0.665	1.620	8.481	10.812
(Former)	1.138	0.641	1.669	7.149	9.554
(Latter)	0.955	0.683	1.583	9.806	12.064

Table 8 Simulation results on a static network

4.4 Simulation 3 Discovery on a Dynamic Network

One of the advantages of Peer to Peer Networks is Robust to dynamic network changes due to broadcast based discovery. We show that our proposed discovery has also a great potential adaptive to dynamic networks through the following simulations.

First of all, we define dynamic of network, Dynamic Rate.

4.4.1 Dynamic Rate

First, we define dynamic rate on dynamic networks. In this simulation, dynamic on our assumption model is brought by Cyber Entity's Migration Distance (In this simulation we remove any other factor to lead dynamic networks to avoid complicated problems.)

The change of relationship partners before and after one cycle is an index of dynamic networks because a Cyber Entity will do sensing everytime it receives a query on our assumption model. Therefore, we define dynamic rate as the average time it keeps a relationship partner.

Therefore, Dynamic Rate is the following;

$$\text{Dynamic Rate} = \frac{(\text{Observed staying time of a relationship partner})}{\text{The number of observed relationship partners.}}$$

In this simulation, Cyber Entity's Migration distance is only a parameter to make the network dynamic. Table 9 shows the relation between Migration Distance and Dynamic Rate

Migration Distance	Dynamic Rate
Migration 2	37.24
Migration 3	32.49
Migration 4	23.33
Migration 5	17.08
Migration 6	14.50
Migration 10	7.75

Table 9 Relation between Migration Distance and Dynamic Rate

In the following simulation, we use dynamic network at Migration 3 and Migration 6.

4.4.2 Community formation

To see community formation, we measured sum of keyword strength on all relationships at search originator. The results are given in Fig.11, 12, 13 and 14. s. From Fig.11 and 12 some communities might be created on the dynamic network. From Fig.11 we can see that total keyword strength became 0 frequently. As keyword strength on a relationship will be 0 when its relationship disappears, the reason is that a community by Rate consists of single root. Therefore, we can say that Rate forwarding is a kind of unicast as we can expect.

On the higher dynamic network, we can see from Fig.13 and 14 that large communities are hardly generated and most small communities would disappear in a moment. Community is, we can expect, valid only to the extent of dynamic. Community can't be generated on the higher dynamic network. It is not bad for the concept of the community to have such a feature, because it is useless to have historical data on such a high dynamic network as a community can't be generated.

4.4.3 Discovery Range

To estimate discovery range of our proposed scheme, we measured the number of entities required to discover at least one keyword entity. The results are given in Table 11 and 12.

As the network is more dynamic, relationship network changes more dynamically. This means size of community will change according to the dynamic extent because community consists of relationship network.

From the case of Rate in Table 11, we can see that the number of entities decreased. This means that community keeps entities from useless forwarding. So a query is forwarded to a solution by unicast. The number of entities receiving a query at Migration 6 is larger than at Migration 3. This doesn't mean that discovery range is larger. When a search query doesn't reach any keyword entity, the results are carried over to the next discovery. For that, the results become worse apparently.

At Migration 6, both schemes Rate and Sigmoid give us similar results. These also mean that both schemes are closer to a random search on such a high dynamic network.

<i>Migration 3</i>	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	3.400	25.733	30.275
<i>(Cycle 1-500)</i>	5.868	19.418	29.148
<i>(Cycle 501-1000)</i>	1.931	31.784	31.402

Table 10 the number of entities receiving a query in one cycle

<i>Migration 6</i>	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	13.023	14.462	20.926
<i>(Cycle 1-500)</i>	14.237	12.285	20.338
<i>(Cycle 501-1000)</i>	12.067	17.016	21.563

Table 11 the number of entities receiving a query in one cycle

4.4.4 Discovery Efficiency

To see effective discovery, we measured the average number of entities received a query to discovery one keyword entity. The results are given in Table 13. At Migration 3, we can say that our proposed schemes, Rate and Sigmoid, give us more effective discovery than broadcast provides. In the latter part (Cycle501-1000), we can see the great improvement from Table 13. This means there are few duplicated forwarding because of unicast.

<i>Migration 3</i>	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	2.565	3.999	5.681
<i>(Cycle 1-500)</i>	4.093	4.166	5.167
<i>(Cycle 501-1000)</i>	1.656	3.839	6.194

Table 12 Discovery efficiency at Migration 3

4.4.5 Overhead

To estimate overhead of query duplication, we measured the times of receiving the same query in a cycle. The results are given Table 14. We can see that Rate and Sigmoid cut down the overhead even on a dynamic network.

<i>Migration 3</i>	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	1.195	2.099	3.020
<i>(Cycle 1-500)</i>	1.371	2.025	2.479
<i>(Cycle 501-1000)</i>	1.091	2.169	3.561

Table 13 Overhead at Migration 3

4.4.6 Quality of Query Hits

To confirm validness of our proposed scheme, we measured all goodness of query hits a search originator received and acquired the average. The results are given in Table 13. And Fig. 15, 16, 17 also show the results. Though Fig.15 shows there are many slits, both of schemes give us good results.

<i>Migration 3</i>	<i>Rate</i>	<i>Sigmoid</i>	<i>Broadcast</i>
<i>Average(Cycle 1 – 1000)</i>	0.842	0.744	0.555
<i>(Cycle 1-500)</i>	0.851	0.786	0.559
<i>(Cycle 501-1000)</i>	0.837	0.706	0.551

Table 14 the average Quality of Query Hits at Migration 3

4.4.7 Summary

To see comparative performance of our discovery schemes with broadcast, we calculated previous measurements treated broadcast's measurement as 1. The results are given in Table 15 and Table 16 also shows the average maximum hops to a keyword entity and maximum hops in a discovery. From the results, our proposed discovery schemes give us good results to a certain extent dynamic network. That is, our scheme is robust to a certain extent dynamic network

<i>Migration 3</i>	<i>Discovery Efficiency</i>	<i>Overhead</i>	<i>Quality of Query Hit</i>	<i>Max Hops to solution</i>	<i>Max Hops</i>
<i>Rate</i>	0.452	0.396	1.517	1.845	2.118
(Former)	0.792	0.553	1.523	2.844	2.941
(Latter)	0.267	0.306	1.519	1.250	1.473
<i>Sigmoid</i>	0.704	0.695	1.342	5.337	7.041
(Former)	0.806	0.817	1.406	4.317	6.006
(Latter)	0.620	0.609	1.281	6.314	8.054

Table 15 Simulation results on a dynamic network at Migration 3

<i>Migration 6</i>	<i>Discovery Efficiency</i>	<i>Overhead</i>	<i>Quality of Query Hit</i>	<i>Max Hops to solution</i>	<i>Max Hops</i>
<i>Rate</i>	0.945	0.812	1.013	3.504	3.885
(Former)	2.129	0.702	1.472	3.704	3.935
(Latter)	0.544	0.896	0.720	3.347	3.838
<i>Sigmoid</i>	0.750	0.559	1.475	3.475	4.828
(Former)	1.464	0.580	1.990	3.126	4.605
(Latter)	0.491	0.536	1.134	3.884	5.060

Table 16 Simulation results on a dynamic network at Migration 6

5 Conclusions and future work

In this paper, we proposed new discovery scheme for peer to peer networks. The proposed discovery scheme is based on the community and the probabilistic forwarding of a query packet, and it solves problems with the current peer to peer discovery schemes (i.e., namely, the difficulty of specifying a right TTL value, receiving too many query hits with low user level satisfaction, vulnerable to dynamic changes in the network). We have examined various characteristics of the proposed discovery mechanism through simulations and verified that the proposed discovery scheme solves the problems with the current discovery schemes.

Through simulations, we observed that the efficiency of the proposed discovery scheme depends on various system parameters such as Gnutella, Freenet, and Neuro Grid. Depending on parameter values, in some simulation scenarios, we observed that a query packet only propagates over a small number of hops (such as five hops) over cyber entity connectivity, and if the target cyber entity (or cyber entities) lie far from the query originator cyber entity, it takes a large number of search tries before the target cyber entity (cyber entities) is (are) successfully found. Improving the probabilistic forwarding so that a query can hop over a large distance is one of a key future research topics with the proposed discovery algorithm.

Lastly, we believe that the proposed concept of community can be applied to other areas, not only the peer to peer discovery.

Acknowledgements

Before everything, I would like to express my gratitude for Professor Tastyua Suda of University of California, Irvine. He gave me a chance to go to UCI. We had a lively discussion about this thesis at UCI. His words impressed themselves on my memory. He also gave me some chances to perform a talk on this paper at various place, such as DARPA, Fujitsu, Hitachi, NTT and so on. Especially, DARPA Meeting strongly impressed upon me at Washington D.C. I am deeply indebted to Professor Masao Fukushima for his earnest guidance and his gentle kindness. He allowed me to go to U.S. Especially I respect him from his personality. I am very grateful to Associate Professor Tetsuya Takine from the bottom of my heart. His intelligence was my motivative to enter our laboratory. Moreover, he gave us very exciting seminars on queueing theory, which were . In the seminar he had taught us various things. I would like to thank all the members in Professor Fukushima's Laboratory for cheering me up. In addition, I would like to thank all the members Ariffin Yahaya, Yi Pan, Michael Moore, Jun Suzuki and Hikaru Yamada in Netgroup at UCI. Thanks to them, I could live life to the fullest. Especially, I would like to show a special deal of appreciation to Mr Keita Fujii for helping me to lead my life in U.S. and all the support he has given to me. In addition, I want to give my thanks to NTT Mirai Lab. Membors Masato Matsuo, Tomoko Itao, Tetsuya Nakara and Miyuki Imada. Thanks for Yutaka Okaie he reminded me of Hiroshima soul during my visiting to U.S. I would like to appericite Naomi Yamashita greatly. Her E-mail during her working time brought me surprised and delightful. And give my sincere thanks for Sakaya and her sister Madoka Yamazoe, they encouraged me everytime.

Lastly, I dedicate this paper to my family with my honest gratitude.

Reference

- [1] Gnutella Wego Website, <http://gnutella.wego.com/>
- [2] Freenet Website, <http://www.freenet.org/>
- [3] NeuroGrid Project Website, <http://www.neurogrid.net/>
- [4] M. Weiser, "The Computer for the Twenty-First Century," *Scientific American*, pp. 94-101, 1991.
- [5] T. Suda, T. Ito, T. Nakamura and M. Matsuo, "Adaptive Networking Architecture for Service Emergence," the Trans. Inst. Electronics Commun. Engineers of Japan, Invited Paper, Vol. J84-B, No. 3, pp. 310-320, 2001.
- [6] T. Suda, "The Bio-Networking Architecture: A Scalable and Self-Organizing Network Architecture based on Biological Concepts," Santa Fe Institute Workshop on The Internet as a Large-Scale Complex System, 2001.
- [7] Anthill Website, <http://www.cs.unibo.it/projects/anthill/>
- [8] F. Schwartz, "Discovering shared interests using graph analysis," *Communications of the ACM*, Vol. 36, No. 8, pp. 78-89, 1993.
- [9] J. Ritter, "Why Gnutella Can't Scale. No, Really," Website, <http://www.darkridge.com/~jpr5/doc/gnutella.html> 2001.
- [10] N. Minar, M. Gray, O. Roup, R. Krikorianm and P. Maes, "Hive: Distributed Agents for Networking Things," *Proc. of ASA/MA*, 1999.
- [11] The BNA project Website, <http://netresearch.ics.uci.edu/bionet>
- [12] S. Milgram, "The small world problem," *Psychology Today*, vol. 1, No. 1, pp. 60-67, 1967.
- [13] S. Sugizaki and T. Suda, "Evolution Simulations for the Bio-Networking Architecture," Inst. Electronics Commun. Engineers of Japan Next Generation Networks workshop, 2001.
- [14] J. Autstin, "How to Do Things with Words," Clarendon Press, 1962.

Appendix

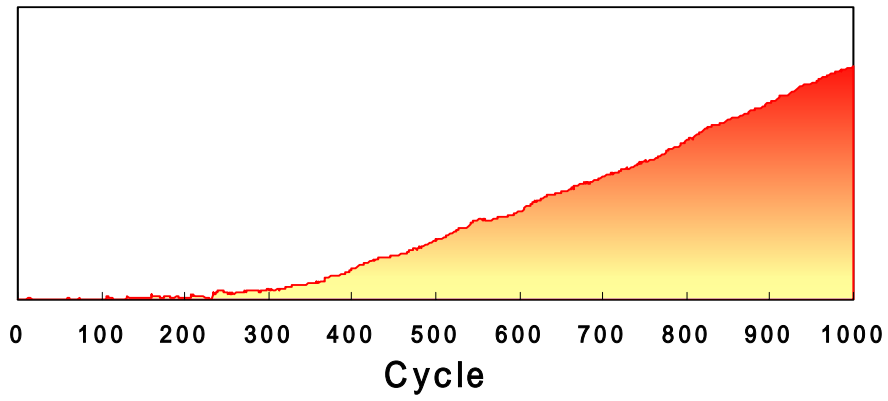


Fig. 7 Total Keyword Strength (Rate Static)

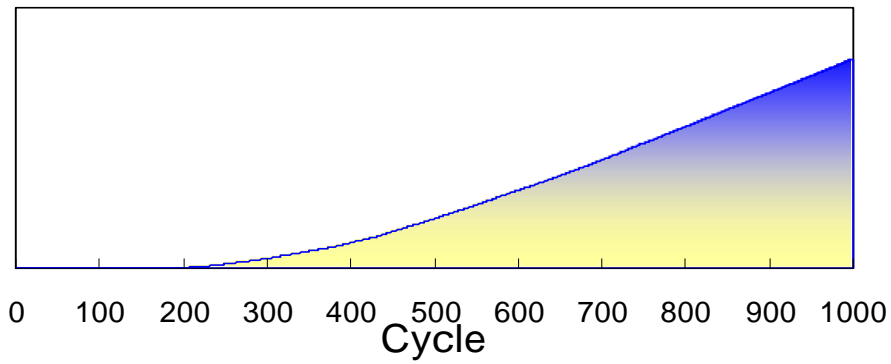


Fig. 8 Total Keyword Strength (Sigmoid Static)

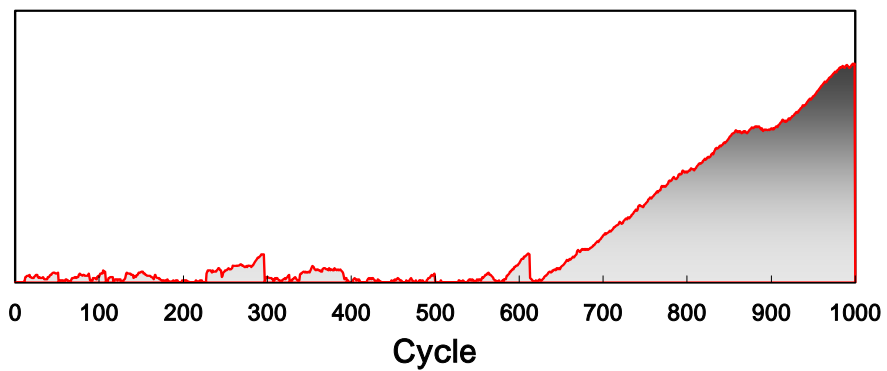


Fig. 9 Total Keyword Strength at Migration 3 (Rate)

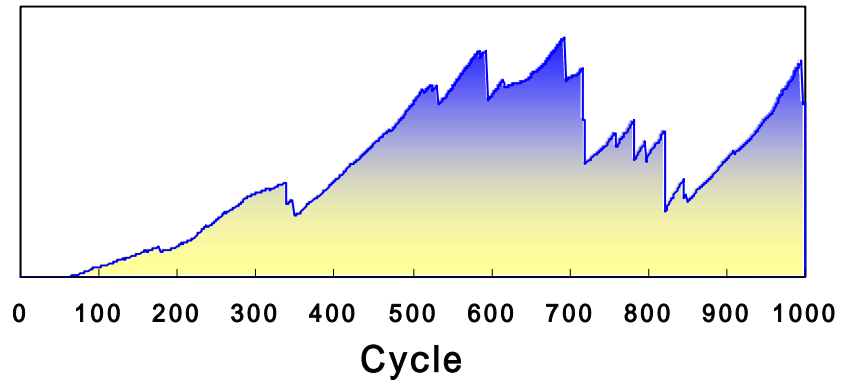


Fig. 10 Total Keyword Strength at Migration 3 (Sigmoid)

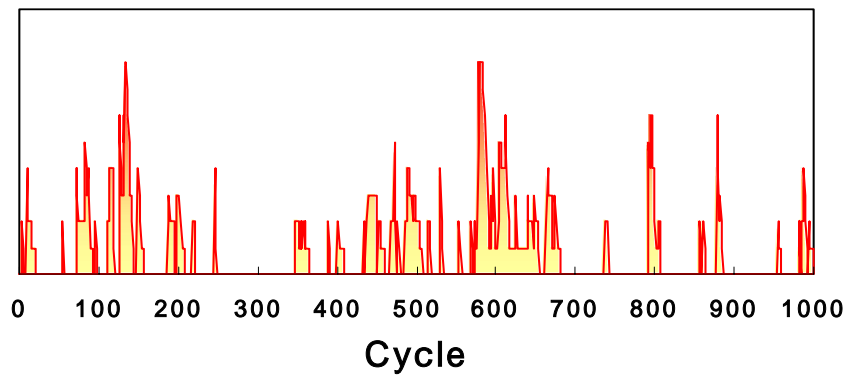


Fig. 11 Total Keyword Strength (Rate Migration 6)

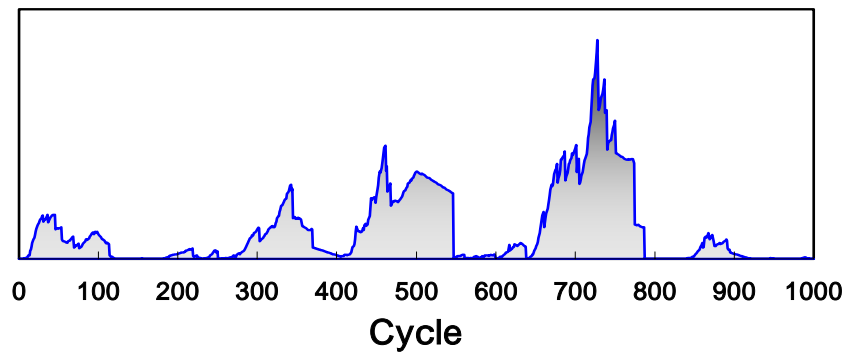


Fig. 12 Total Keyword Strength (Sigmoid Migration 6)

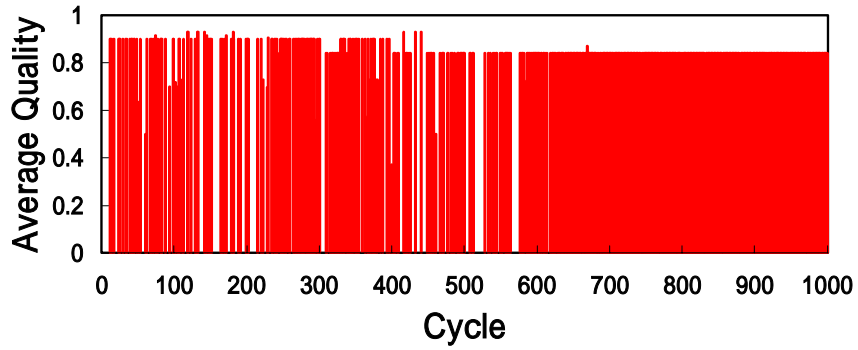


Fig. 13 The average quality of query hits at Migration 3 (Rate)

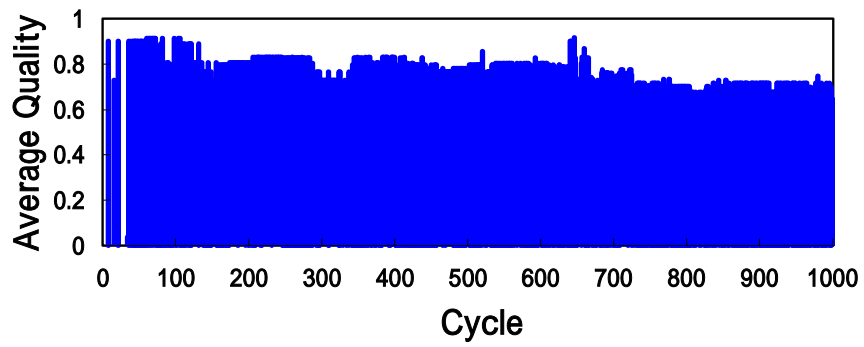


Fig. 14 The average quality of query hits at Migration 3 (Sigmoid)

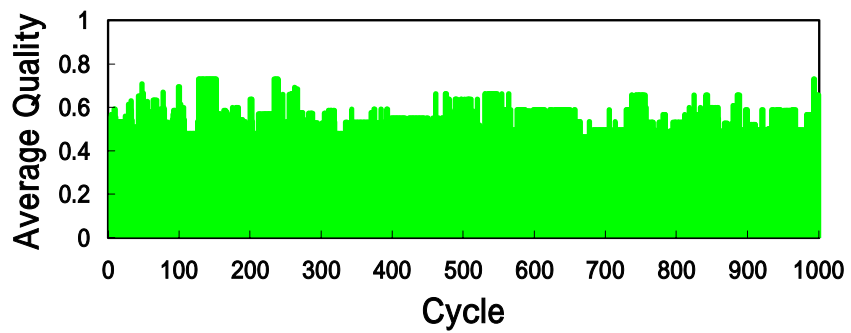


Fig. 15 The average quality of query hits at Migration 3 (Broadcast)