Peer-to-Peer in Metric Space and Semantic Space

Hai Zhuge, *Senior Member*, *IEEE*, and Xiang Li

Abstract—This paper first proposes three improved gossip mechanisms by mapping links into metric space and dynamically adapting the number of selected neighbors to disseminate messages. Experiments and comparisons show that these mechanisms can improve the performance of gossip in peer-to-peer (P2P) networks. This is the effect of mapping a network into a metric space that differentiates nodes and links according to linking characteristics and controlling local information flow with knowing such differences. A further study about query routing on P2P semantic link network shows that mapping a network into a semantic space can also improve the performance. An intrinsic rule is found by experimental comparisons and analysis: The performance of a P2P network can be improved by designing an appropriate mapping from the network into metric space or semantic space. A general framework for networking with metric space and semantic space is suggested.

Index Terms—Gossip, metric space, peer-to-peer, rank, semantics, semantic link network.

1 Introduction

1.1 Peer-to-Peer (P2P) Network

With pervasive deployment of computers and networks, P2P is receiving more and more attention due to its decentralization, self-organization, scalability, fault resilience, ad hoc connectivity, low cost of ownership, and anonymity [2], [4]. Nodes (peers) can join and leave casually and the cost of ownership is shared among peers. P2P networks and their applications need to know this ad hoc nature and handle joining and departing behaviors.

Structured P2P networks like Content-Addressable Network (CAN), Chord, Pastry, and Tapestry [28], [32], [34], [38] assign each file a unique key and build a Distributed Hash Table (DHT) that maps each key onto a specific peer. The whole network is controlled by the structured DHT and supports only the exact match. Such a network enables efficient search and guarantees accurate locating of resources, but its strict structure leads to comparatively less dynamicity and high maintenance cost. As an improvement, a decentralized algorithm is introduced to support parallel construction of structured overlay networks based on a recursive bisection scheme [1]. In recent years, establishing various indexing overlays on structured P2P network to support advanced applications has drawn attention [42].

In unstructured P2P networks like Gnutella (http://rfc-gnutella.sourceforge.net) and KaZaA (www.kazaa.com), peers are totally self-organized and resources are randomly placed without any restrictions. Such a network topology is easy to maintain and is remarkably robust against accidental failures, but locating resources is inefficient because it cannot obtain global information.

• *The authors are with the Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. E-mail: zhuge@ict.ac.cn.*

Manuscript received 17 May 2006; revised 5 Dec. 2006; accepted 4 Jan. 2007; published online 1 Mar. 2007.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0259-0506.
Digital Object Identifier no. 10.1109/TKDE.2007.1035.

1.2 Gossip on Unstructured P2P Networks

Gossip mechanisms are also called epidemic algorithms for application-level multicast. They have attractive scalability, reliability, and graceful degradation properties. Every node that receives a message randomly selects a certain number of nodes from its neighbors to disseminate the message. The use of redundant information guarantees the reliability in such cases as node crash and network package lost. These approaches scale well as the load of nodes grows logarithmically in comparison with the number of nodes in the network.

Gossip-based mechanisms disseminate messages from one node to randomly selected neighbors just as an infected individual passes a virus onto other people that he/she comes across [3]. The inherent scalability of gossip-based mechanisms makes them suitable for disseminating information in large-scale P2P networks. Meanwhile, they are resilient to changes in the underlying network topology and participants' failure. Moreover, these algorithms are easy to implement and inexpensive to run and they impose constant loads on participants. The throughput is stable over a relatively long period and overheads are flat, predictable, and can be balanced with information about network topology [36]. Compared with general flooding approaches, the gossip mechanisms are more scalable in terms of load imposed on the networks and participating peers when system size grows into hundreds or even thousands of peers. The deterministically constrained flooding approach can lower overhead when there are a small number of failures. However, its reliability degrades faster than that of gossip when failure rates increase [19]. Being easy to deploy, robust, and highly resilient to failures, gossip-based mechanisms can effectively disseminate information in large P2P systems deployed on Internet or ad hoc networks.

1.3 Existing Gossip Mechanisms and Applications

A gossip mechanism that scales well and provides timely failure detection is introduced in [30]. It can be used to

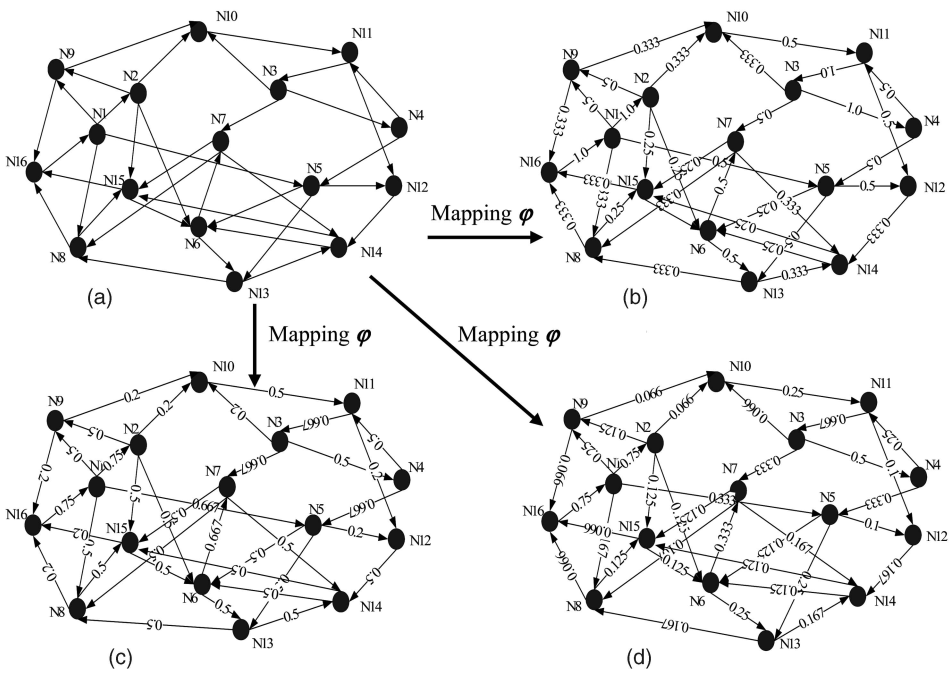

Fig. 1. (a) An example of gossip network. (b) Mapping result by considering incoming links. (c) Mapping result by considering outgoing links. (d) Mapping result by considering both incoming and outgoing links.

discover and leverage the underlying network topology for improved resource utilization and to handle partition failures in combination with other mechanisms.

Bimodal multicast is reliable as it can be rigorously quantified and guarantees stability that might be needed in the Internet, radio, television, and conferencing applications [7]. The newscast model is for communication in large agent-based distributed systems. It uses the same epidemic approach to implement membership management and information dissemination [37].

Scalable management and self-organizational capabilities are central requirements of large-scale highly dynamic and distributed applications. A system offers a scalable way to track system state as it evolves [29]. Its combinational features could solve a variety of management and self-configuration problems.

The P2P membership protocol Scalable Membership Protocol (SCAMP) operates in a decentralized manner and provides each member with a partial view of the group membership [12]. The protocol is self-organized in the sense that the partial views naturally converge to the size required to support a gossip algorithm reliably.

Epidemic information dissemination concerns the following key issues [11]:

1. membership maintenance—how peers get to know each other and how many others in the system do the peers need to know,
2. network awareness—how to connect peers by making use of actual network topology to ensure acceptable performance,
3. buffer management—which information should be dropped at a peer when its storage buffer is full, and
4. message filtering—how to take into account the actual interest of peers and decrease the probability that peers receive and store information of no interest to them.

Epidemic algorithms have been applied to such areas as failure detection, data aggregation, resource discovery, database replication, resource monitoring, information dissemination, and e-science [10], [40].

Now, we start from investigating three mechanisms to improve the performance of existing gossip mechanisms by mapping into metric space.

2 Mapping Gossip into Metric Space

This section shows how routing performance is improved in metric space.

2.1 Mapping Mechanisms

2.1.1 The Networking Model

In gossip mechanisms, each node that receives a message randomly selects some of its neighbors to disseminate the message. Therefore, some richly connected nodes would receive too many copies of the same message; meanwhile, some poorly connected nodes would only receive few copies or may even be isolated. As Fig. 1a shows, node N10 can receive messages from nodes N2, N3, and N9, whereas it can only send messages to node N11. Some nodes like node N2 are poor in receiving messages. As a result, node N2 will be isolated with high probability in the following situations: Node N1 is crashed, the message is dropped, and node N2 is not chosen as the next target to forward messages when N1 receives a message. At the same time, nodes N14, N15, and N16 are relatively rich in receiving messages. They could receive messages from multiple neighbors so that the probabilities of receiving messages are high. Our idea is to allow nodes to know local linking characteristic and establishing appropriate mappings from the network into metric space.

Distributed algorithms and protocols can be localized if nodes make decisions based solely on knowledge of one-hop or two-hop neighbors [27]. Requiring relatively less global information is one of the important needs for scalable protocols. Our mapping mechanisms use linking information of each node's one-hop neighbors. The following definitions will be used in the discussion:

1. $fanout$ is the number of neighbors one node selects to disseminate when it receives a message.
2. TTL (Time To Live) is the iterative rounds for a message to disseminate.
3. $outView(i)$ is the neighbors that node i can send messages to.
4. $inView(i)$ is the neighbors that node i can receive messages from.
5. $link_{ij}$ is the link pointing from node i to node j.
6. φ is the mapping from the network into metric space. A distance d exists between any two points in metric space and, for any three points x_1, x_2, and x_3 in metric space, we have
$$d(\mathrm{x}_1, \mathrm{x}_3) \leq d(\mathrm{x}_1, \mathrm{x}_2) + d(\mathrm{x}_2, \mathrm{x}_3).$$
This focuses on real number space.
7. $\varphi(link_{ij})$ is the rank of $link_{ij}$.
8. p_{ij} is the probability that node j is selected as the next target node to transmit the message when node i receives a message and decides to forward it, $p_{ij} = \varphi(link_{ij}) / \sum_{j \in outView(i)} \varphi(link_{ij})$.

The *message disseminating process* is given as follows: When node i receives a message, it checks the TTL; if $TTL > 0$, i selects $fanout$ neighbors from its $outView$ randomly with different probabilities p_{ij}; then, TTL is decreased by 1; else, the message propagating process is terminated. In the random selection process, the link with larger rank in real number space obtains higher probability to transmit the message.

In the following, three different mapping mechanisms are proposed for increasing the probability of poorly connected nodes to receive messages and decreasing the number of messages sent to richly connected nodes.

2.1.2 Mapping Mechanism Based on Incoming Links

Nodes like node N2 with a small number of incoming links have relatively little probability of receiving propagated messages. One way to increase the opportunity for node N2 to receive messages is to raise the probability that N2 is selected as the target node when node N1 receives a message. For this, the ranks of links pointing to the nodes like N2 should be increased. Therefore, the mapping mechanism considering incoming links is designed as follows:

$$\varphi(link_{ij}) = \begin{cases} \alpha_1 / |inView(j)| & |inView(j)| > 1 \\ 0 & |inView(j)| = 0, \end{cases}$$

and $\alpha_1 > 0$, where $|inView(j)|$ is the number of nodes in $inView(j)$.

Using this mapping mechanism, we obtain Fig. 1b from Fig. 1a, with $\alpha_1 = 1.0$. In the figure, the rank of $link_{N1,N9}$ is 0.5, and the rank of $link_{N1,N8}$ is 0.333. When node N1 receives a message from node N16, it will select $fanout$ nodes randomly from its outgoing neighbors, that is, nodes N2, N5, N8, and N9, to disseminate the message. In each random selection process, N2 will be chosen with probability $1.0/(1.0 + 0.5 + 0.333 + 0.5) \approx 0.43$. Similarly, nodes N5, N8, and N9 can be selected with approximate probability 0.21, 0.14, and 0.21, respectively. Note that, without rank, N2, as well as N5, N8, and N9, will be selected from node N1's $outView$ with the uniform probability $1.0/4 = 0.25$. Therefore, the poorly connected nodes' opportunity of receiving messages can be improved. The richly connected nodes could receive fewer redundant messages and poorly connected nodes could receive more messages.

2.1.3 Mapping Mechanism Based on Outgoing Links

Greedy algorithms always take the best immediate or local solution in problem solving. They find the globally optimal solution for some optimization problems but may find less-than-optimal solutions sometimes. The purpose of gossip is to disseminate messages to peers in the system. Therefore, in the gossip process, sending a message to peers with more outgoing neighbors should have greater potential to reach the purpose. When a peer receives a message, it greedily selects target peers from its $outView$. That is, the larger the neighbor's outgoing degree is, the higher the probability it is selected as the target node. The mapping mechanism based on nodes' outgoing links can be designed as follows:

$$\varphi(link_{ij}) = \begin{cases} \alpha_2 - \alpha_2 / |outView(j)| & |outView(j)| > 1 \\ \beta_1 & |outView(j)| = 1 \\ \beta_2 & |outView(j)| = 0, \end{cases}$$

and $0 < \beta_2 < \beta_1 < \alpha_2/2$, where $|outView(j)|$ is the number of nodes in node j's $outView$. The value of $\alpha_2 - \alpha_2/|outView(j)|$ is within $[\alpha_2/2, \alpha_2]$ when $|outView(j)| > 1$. When $|outView(j)| = 1$, we set it as β_1 within $(0, \alpha_2/2)$ since $\varphi(link_{ij})$ should not be zero. When $|outView(j)| = 0$, the smaller value β_2 is set since we could not use the uniform function. Taking $\alpha_2 = 1.0$, $\beta_1 = 0.2$, and $\beta_2 = 0.1$, we obtain Fig. 1c from Fig. 1a.

2.1.4 Mapping Mechanism Based on Both Incoming and Outgoing Links

The mapping mechanism based on both incoming links and outgoing links can be designed as follows:

$$\varphi(link_{ij}) = \begin{cases} \alpha_1(\alpha_2 - \alpha_2/|outView(j)|)/|inView(j)| & |inView(j)|>0, |outView(j)|>1 \\ \beta_1/|inView(j)| & |inView(j)|>0, |outView(j)|=1 \\ \beta_2/|inView(j)| & |inView(j)|>0, |outView(j)|=0 \\ 0 & |inView(j)|=0, \end{cases}$$

where $\alpha_1 > 0$, $0 < \beta_2 < \beta_1 < \frac{\alpha_2}{2}$, and $|inView(j)|$ and $|outView(j)|$ are the number of nodes in node j's $inView$ and $outView$, respectively. Taking $\alpha_1 = \alpha_2 = 1.0$, $\beta_1 = 0.2$, and $\beta_2 = 0.1$, we obtain Fig. 1d from Fig. 1a.

2.2 Peer Management

2.2.1 Peers Join Management

When a new node p joins by contacting node c, node p forwards a *join* message including its ID to c. Node c randomly selects t number of nodes in its $outView$ according to the rank of link between them (the larger rank it has, the higher probability it will be selected) and forwards the *join* message to them.

The nodes (including node c) receiving the *join* message first decide whether to add the ID in the message to their $outView$s or $inView$s according to their corresponding free buffer size. If its (for example, node r's) $outView$'s free size is larger than its $inView$ free size, the node ID is added to its $outView$, and it sends *out* message to p; else, if r's $outView$'s free size is less than its $inView$ free size, the ID is added to its $inView$, and it sends *in* messages to p. If the two have the same free buffer size, one of them is chosen randomly and sends *out* or *in* messages back accordingly. Then, node r adjusts the links' ranks relating to it according to our mechanisms. If r's membership buffer size is full, node r does the similar thing iteratively as done by c, and the TTL decides whether to end the iteration.

Once node p receives the message, it will check the message type; if the message is *out*, the node adds the node ID in the message to its $inView$; otherwise, it adds the ID to its $outView$. Node p adjusts the links' ranks accordingly.

2.2.2 Peers Departure Management

When node r wants to leave the system, the following mechanism is adopted to keep the nodes in r's $inView$ with those in its $outView$ connected: For each node (take node s for example) in its $outView$, node r selects one of the node IDs (q for example) from its $inView$ randomly and then forms a *failure* message including q and forwards it to s. When s receives the message, it substitutes r's ID (r) with q in its $inView$, adjusts the links' ranks accordingly, and then forwards a message with r and q to the system. When q receives the message, it will update r with q in its $outView$. Therefore, withdrawal of a pivot node will not lead to the partition of the whole network. If a node crashes without following the protocol, the neighbor nodes could maintain their memberships by ways like interchanging the states periodically [30].

2.3 Average Traffic Analysis

Kermarrec et al. consider the graph of n nodes where the edge between each pair of nodes exists with probability $[\log(n) + c + o(1)]/n$. In this prerequisite, the probability that the graph is connected goes to $exp(-exp(-c))$, where c is a constant. Moreover, the target could be reached by defining the appropriate sizes of the $inView$ and $outView$ of nodes in the system. For networks constructed this way, the required $fanout$ has a sharp threshold at $\log(n)$ [16]. Therefore, the gossip systems with size n will have a promising effect when the $fanout$ value is set to be around $\log(n)$. Taking a gossip system of 1,000 nodes for example, the $fanout$ value will be ideal when it is around 3.

The term *average traffic* means that the average number of messages per node received after a message dissemination process is terminated. The total number of messages disseminated in the process is

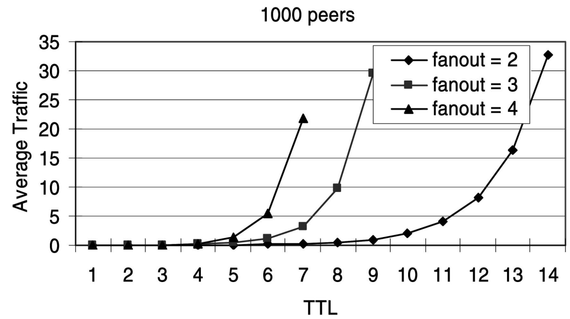

Fig. 2. Average traffic distribution with different $fanout$s.

$$fanout^1 + fanout^2 + \ldots + fanout^{TTL-1} + fanout^{TTL}$$
$$= fanout(fanout^{TTL} - 1)/(fanout - 1),$$

then the average traffic will be

$$fanout(fanout^{TTL} - 1)/(n(fanout - 1)).$$

With $fanout$ around $\log(n)$, the average traffic should be $\log(n)(\log^{TTL}(n) - 1)/(n(\log(n) - 1))$ approximately. Fig. 2 intuitively demonstrates the relationship among the average traffic, $fanout$, and TTL when the number of nodes is 1,000. It shows the reasonable range of parameter TTL with respect to different $fanout$s.

2.4 Message Propagation Analysis

With reference to the epidemic model [15], we analyze the information dissemination process using the following notations:

1. $s(t)$ is the number of nodes that have not received the message at time t.
2. $i(t)$ is the number of nodes that have received the message at time t.
3. n is the total number of nodes in the system.
4. k is the infective factor which influences the speed of information dissemination.
5. t is the time since the process of information dissemination starts; here, TTL is the metric.

As there are only two types of nodes, one has received the message and the other has not, we have

$$s(t) + i(t) = n. \tag{1}$$

Assume that the increasing rate of the number of nodes that have been "infected" by the message is in direct proportion to $fanout$, $s(t)$, and $i(t)$. Therefore, we obtain

$$\frac{di(t)}{dt} = k \times fanout \times s(t) \times i(t). \tag{2}$$

Considering the initial condition $i(0) = i_0$, we resolve the differential equation (2) in combination with (1):

$$i(t) = \frac{n}{1 + (\frac{n-i_0}{i_0})e^{-fanout \times k \times n \times t}}.$$

We get the first and second differential of $i(t)$ as follows:

$$i'(t) = \frac{fanout \times n^2 \times c \times k \times e^{fanout \times n \times k \times t}}{(1 + c \times e^{fanout \times n \times k \times t})^2},$$

where $c = i_0/(n - i_0)$, and

$$i''(t) = fanout \times k \times i'(n - i) - fanout \times k \times i \times i' = fanout \times k \times (n - 2\,i)i'.$$

Letting $i'' = 0$, we get the inflection point $i = n/2$. That is, when $t = t_1 = \frac{1}{fanout \times k \times n}\ln(\frac{n-i_0}{i_0})$, i' obtains its maximum value, which is the point that $i(t)$ grows fastest.

When time t and n are fixed, the increase of $fanout \times k$ will result in the decrease of $e^{-fanout \times k \times n \times t}$ and, as a result, $i(t)$ will increase. Hence, increasing the $fanout$ value will accelerate the process of information dissemination. Our mapping mechanisms make the information dissemination more dispersed; therefore, the message will reach more nodes than the one without considering rank within the same time range. Therefore, the proposed mapping mechanisms positively influence the infective factor k, which could also shorten the time for the network to approach the inflection point.

Similarly, when $fanout$ is fixed, increasing TTL leads to the increase of $i(t)$ eventually until it reaches n.

2.5 Further Study in Probability Space

Considering a network of n nodes, the overlay of the application-level multicast can be generalized as

$$G = (\{1, \ldots, n\}, E, P),$$

where edge $< i, j > \in E$ only when there exists a link from node i to j and, when i gets a message, it selects neighbor j with probability p_{ij}. The probabilities on all links constitute a probability matrix $P = [p_{ij}]$, for each i, $\sum_{\forall j} p_{ij} = 1.0$ and $p_{ij} > 0$ when $< i, j > \in E$.

Let I_t be the number of peers that have been "infected" by the propagated message at time t; ε and τ, respectively, be the probability of message loss and the probability of a peer crash during a running procedure.

The following is the probability that a given peer is reached from an infected peer in the network with n peers and parameter $fanout$:

$$p(n, fanout) = \frac{fanout}{|outView|} \times \frac{|outView|}{n} \times (1 - \varepsilon)(1 - \tau) = \frac{fanout}{n} \times (1 - \varepsilon)(1 - \tau).$$

Let $q(n, fanout) = 1 - p(n, fanout)$ be the probability that a peer is not reached from a given infected peer. The transition process can be expressed in the form of homogeneous Markov chain:

$$p_{ij}(n, fanout) = p(n, fanout)[I_{t+1} = j | I_t = i] = \binom{n-i}{j-i}(1 - q(n, fanout)^i)^{j-i} \times q(n, fanout)^{i(n-j)}.$$

Moreover, the distribution of $I_t (t > 0)$ can be computed recursively in the following way:

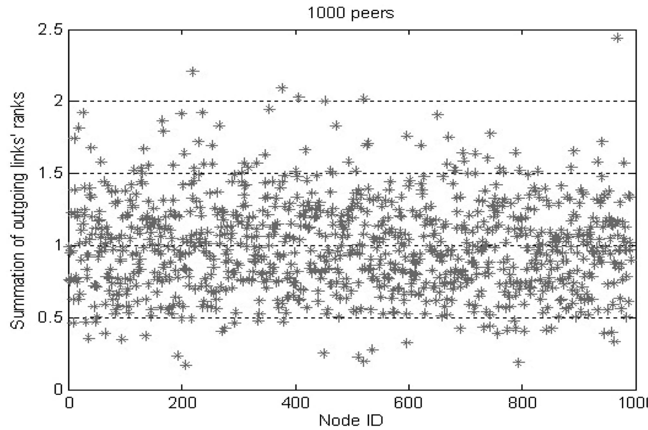

Fig. 3. The distribution of the summation of outgoing links' ranks.

$$p(n, fanout)[I_t = j] = \sum_{i=j/(1+fanout)}^{j} p(n, fanout)[I_{t-1} = i] \times p_{ij}(n, fanout).$$

According to [26], the total rounds $TTL(n, fanout)$ necessary to infect an entire group of size n obey

$$TTL(n, fanout) = \log n \times \left(\frac{1}{fanout} + \frac{1}{\log(fanout)}\right) + c + o(1),$$

where c is a constant.

That is, there exists a trade-off between $fanout$ and TTL in the network of n peers. By increasing the $fanout$ value to some extent, the probabilistic mechanism will evolve to flooding-like ones and, by decreasing the $fanout$ value to 1, the mechanism will change to something like Random Walk [22]. When deploying suitable $fanout$, the TTL will turn out an acceptable value, which will reduce the total cost of the network.

The probabilistic mechanism could reduce the network load sharply compared to flooding methods allowing reliable information to disseminate.

2.6 Fanout Adjusting Mechanism

The average size of $fanout$ can be the function of the total number of nodes n in the network ($\log(n)$). However, just as authorities and hubs exist in the self-organized Web [17], there are some nodes richly connected in the self-organized P2P networks. The difference among nodes implies that making each node in the network the same $fanout$ value is not a good strategy. Considering link ranks, we can allow every node to adjust its $fanout$ value dynamically according to the summation of its outgoing links' ranks.

Fig. 3 shows the summation of nodes' outgoing rank distribution in the network of 1,000 nodes. The summation of outgoing links' ranks is clustered between 0.5 and 1.5 and no summation is larger than 2.5. Only a very few nodes are outside the range.

The extreme situation is that there are some nodes having only one neighbor in their $outView$s. When this kind of node receives a message, it should send the message to the target only one time. If the nodes with rich outgoing links could send messages to more neighbors, the performance could be

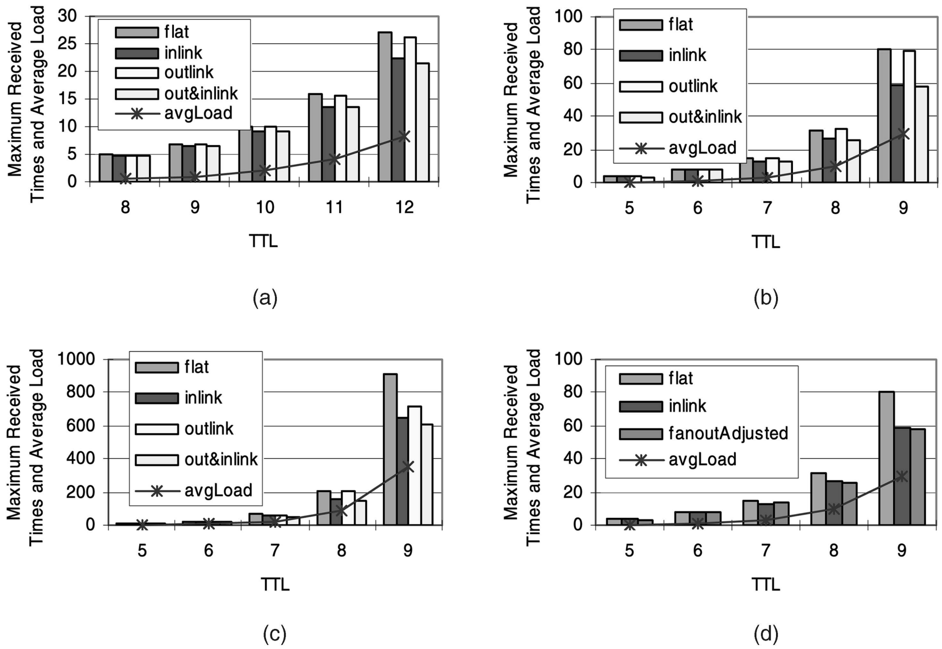

Fig. 4. The distribution of maximum message received times and average load per node in random networks. (a) 1,000 peers, $fanout = 2$. (b) 1,000 peers, $fanout = 3$. (c) 1,000 peers, $fanout = 4$. (d) 1,000 peers, $fanout \approx 3$.

improved by intuition. Therefore, we can allow a node to tune $fanout$ values according to the summation of its outgoing links' ranks. The $fanout$ adjusting mechanism can be set as follows:

$$fanout = \begin{cases} \log(n) - \varepsilon_1 & 0 \leq nol \leq \mu_1 \\ \log(n) & \mu_1 < nol < \mu_2 \\ \log(n) + \varepsilon_1 & \mu_2 \leq nol, \end{cases}$$

where $\varepsilon_1 > 0$ is an integer, $0 < \mu_1 < \mu_2$ are real numbers, and nol represents the summation of outgoing links' ranks.

3 Simulations and Comparisons

To validate the rationality of our mechanisms, we carried out extensive experiments. Each experiment is carried out both on the topology using flat gossip mechanism without considering rank and on the topologies using our mapping mechanisms. The experiments are done on two kinds of directed networks of 1,000 nodes: random networks and random power-law networks. Directed links are added by randomly selecting the start node and the end node. Each experiment with different parameters ($fanout$ and TTL) is carried out 100 times on each network we generated, and the initial node is randomly selected each time. We use the average value of these 100 results to illustrate the feasibility, and the parameters α_1, α_2, β_1, and β_2 are set to be 1.0, 1.0, 0.2, and 0.1, respectively. In each experiment, there is only one node having the message to propagate at the initial state, that is, $i(0) = i_0 = 1$.

We use two performance metrics: 1) comparisons between average network load per node and the maximum times of receiving message (reducing message receiving times leads to the decrease of the number of redundant message that richly connected nodes receive) and 2) the number of nodes which have not received the message.

3.1 Simulation on Random Networks

An epidemic algorithm must make a trade-off between scalability and reliability: Larger views ($inView$ and $outView$) reduce the probability that nodes are isolated or that the network is partitioned, whereas smaller views help the network obtain better scalability. For the random networks, each node's out-degree and in-degree are both 10 in our experiments on the average. Correspondingly, each node's local view on average has 10 neighbors in its $outView$ and $inView$, respectively.

We present the simulation results in Figs. 4 and 5, where the result from the no-rank gossip network is denoted by *flat*, whereas the results from the gossip mechanisms using incoming links, outgoing links, and both outgoing and incoming links are denoted by *inlink*, *outlink*, and *out&inlink*, respectively.

Fig. 4 shows the average network load per node and the distribution of maximum times of messages received according to different parameters. The horizontal axis denotes the parameter TTL and the vertical axis denotes the average maximum times of receiving message in comparison with the average network load during 100 times operation. As Fig. 4a presents, we set the $fanout$ value as 2 uniformly and range TTL from 8 to 12. Figs. 4b and 4c are obtained in a similar way by using different parameters' values. As the figures show, all of the mechanisms exhibit different effects, although they lead to the ascent of maximum received times as the TTL increases. The mechanisms using incoming links, outgoing links, and both outgoing and incoming links gain better effect than the *flat* gossip. The *out&inLink* mechanism obtains the best results, the *inLink* and *outLink* mechanisms take the second and the third place, respectively. Take $fanout = 3$, for example, when TTL approaches 9, about a 27.8 percent improvement

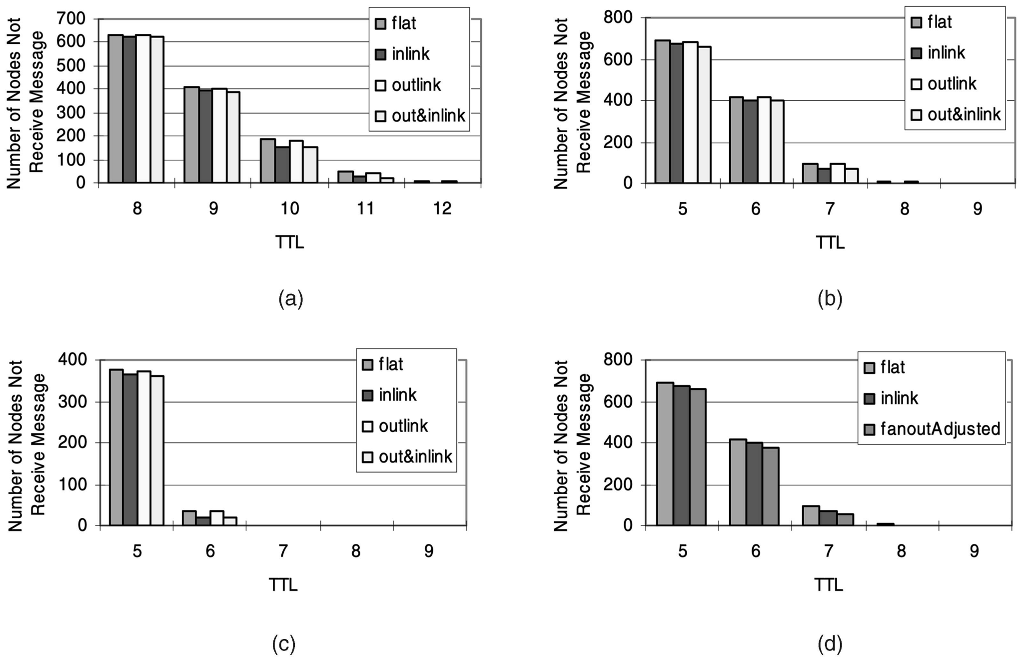

Fig. 5. The distribution of the number of nodes that have not received a message in random networks. (a) 1,000 peers, $fanout = 2$. (b) 1,000 peers, $fanout = 3$. (c) 1,000 peers, $fanout = 4$. (d) 1,000 peers, $fanout \approx 3$.

is gained for the mechanism *out&inlink*, which justifies our approaches.

Experiments were also carried out for verifying the $fanout$ adjusting mechanism. From the result of the 100 times' operation on the network of 1,000 nodes, we get that the average $fanout$ value is approximately 3.019. The result from using the $fanout$ adjusting mechanism is illustrated by *fanoutAdjusted* in Fig. 4d. There is a certain improvement in comparison with the one without $fanout$ adjustment. The link distribution is relatively even in the random networks and this $fanout$ adjusting mechanism will get a better performance when the distribution of the summation of ranks is more dispersed such as power-law distribution. As far as the random network is concerned, the $fanout$ adjusting mechanism can be set as follows:

$$fanout = \begin{cases} 2 & 0 \le nol \le 0.5 \\ 3 & 0.5 < nol < 1.5 \\ 4 & 1.5 \le nol. \end{cases}$$

Another goal of the mapping mechanism is to increase the probability of the poorly connected nodes to receive messages. We are concerned with the number of nodes which have not received the messages during message dissemination. The distribution of the number of nodes that have not received the message is shown in Fig. 5. Compared with previous algorithms, the number of nodes having not received messages evidently decreases after adopting the mapping mechanisms as presented in Figs. 5a, 5b, and 5c. Take $fanout = 3$, for example, when TTL goes to 7, the number of nodes having not received message adopting *out&inlink* mechanism is about 69.8 percent of that using *flat* gossip. In addition, when TTL approaches 8, the metric goes to 0.48 for *out&inlink* and 6.24 for *flat* gossip. In a word, the *out&inLink* mechanism shows the best performance, whereas the *inLink* mechanism takes the second place and the *outLink* mechanism also gains better results than the *flat* gossip mechanism.

The effect of the $fanout$ adjusting mechanism is given in Fig. 5d, which shows that the number of nodes not receiving messages is decreased.

3.2 Simulation on Random Power-Law Networks

Many large networks like the hyperlink network follow the power-law distribution of node degrees [18], [23], [25]. The degree distribution is $p_k = Ak^{-\tau}$, where $A^{-1} = \sum_{k=2}^{k_{\max}} k^{-\tau}$, k is the degree, k_{max} is the maximum degree, and $\tau > 0$ is the exponent of the distribution [33].

The reason for considering power-law networks is that some unstructured P2P networks are characterized by random power-law and heavy-tailed degree distributions [31]. To keep the nodes connected, we adjust the degree from 15 to 150 following the aforementioned distribution with $\tau = 2.0$ in the construction of random power-law networks. For each link, the start node and the end node are selected randomly and, as a result, the random power-law graph is constructed with average degree 35.

The distributions about the maximum times of receiving a message in comparison with the average load per node and the number of nodes which have not received the message in the random power-law network are presented in Figs. 6 and 7, respectively. Take $fanout = 3$ for example, when TTL goes to 7, the number of nodes that have not received message adopting *out&inlink* mechanism is about 49.9 percent of that using *flat* gossip. Moreover, when TTL approaches 8, the metric goes to 1.43 for *out&inlink* and 19.86 for *flat* gossip. As for the maximum received times, when $fanout = 3$ and TTL goes to 9, about 53 percent improvement is obtained for the mechanism *out&inlink* in comparison with the *flat* one, which validates the feasibility of our approaches. The experiment on the random power-law networks shows that the mechanisms based on

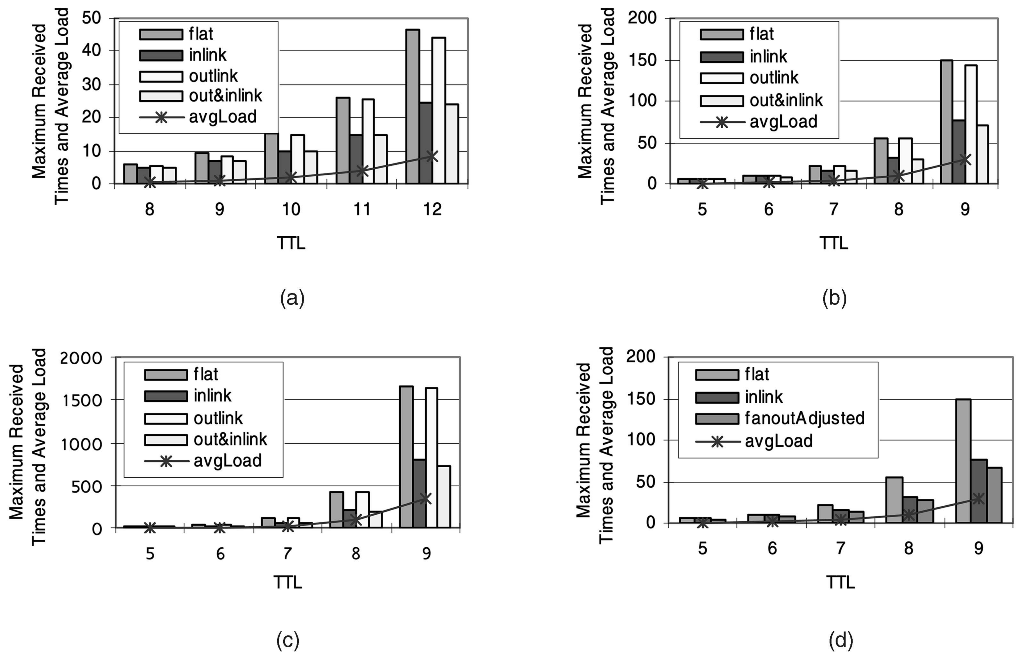

Fig. 6. The distribution of maximum message received times and average load per node in random power-law networks. (a) 1,000 peers, $fanout = 2$. (b) 1,000 peers, $fanout = 3$. (c) 1,000 peers, $fanout = 4$. (d) 1,000 peers, $fanout \approx 3$.

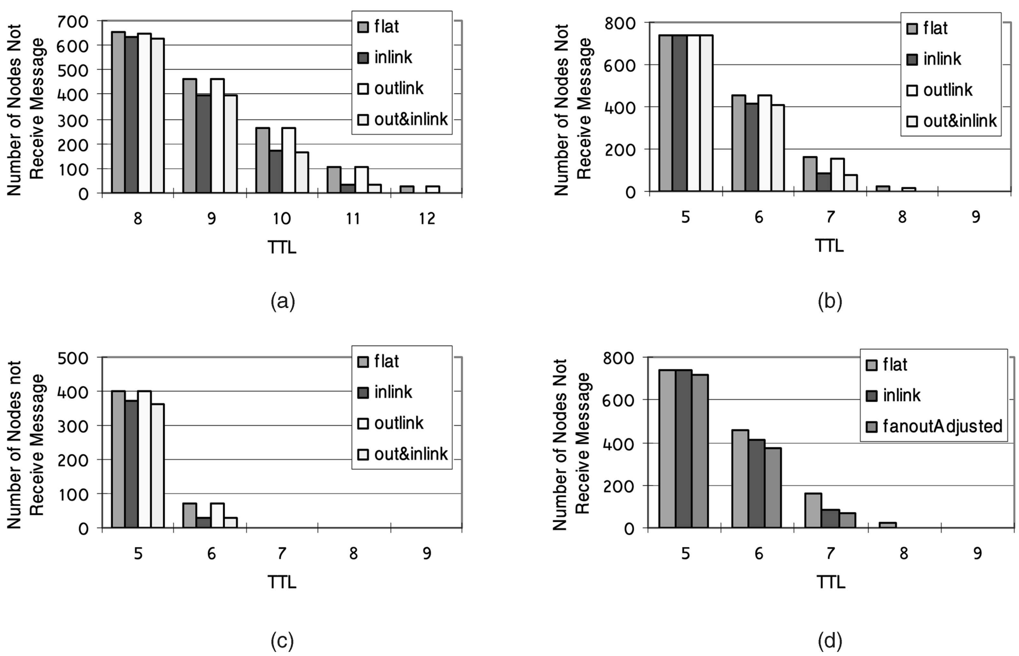

Fig. 7. The distribution of the number of nodes that have not received a message in random power-law networks. (a) 1,000 peers, $fanout = 2$. (b) 1,000 peers, $fanout = 3$. (c) 1,000 peers, $fanout = 4$. (d) 1,000 peers, $fanout \approx 3$.

incoming links, outgoing links, and both outgoing and incoming links perform better than that without using the mapping mechanisms, among which the *out&inLink* mechanism obtains the best results, whereas the *inLink* and *outLink* mechanisms take the second and the third place, respectively.

From the figures, we can see that the *inLink* and *out&inLink* mechanisms achieve considerable effect. Adopting either of the two mechanisms for information dissemination, the number of nodes that do not receive messages and the maximum received times are about half of those without using mapping mechanisms. The *outLink* mechanism also works although not that obviously. Owing to the characteristic that only a few nodes are rich in connection in the power-law networks, the mapping mechanisms work well to direct the information disseminating process.

Due to the uneven distribution of the links in the power-law networks, the $fanout$ adjusting mechanism performs better than that in the random networks. The mechanism also provides a way for the nodes in the P2P network to adjust their $fanout$s autonomously according to their capabilities and linking characteristics.

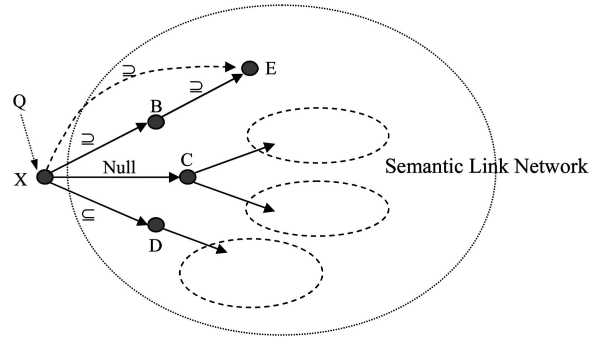

Fig. 8. Query routing strategies in a local semantic link network.

The experiments confirm the feasibility and efficiency of the proposed mapping mechanisms. Essentially, the mapping mechanisms are the effect of mapping a network into a metric space that differentiates links according to local network structure and then adjusting local information flow according to such a difference. This approach is very useful when semantic information is not available in the network.

However, any message has content and the network structure could not reflect the content of nodes.

4 Mapping Gossip into Semantic Space

This section investigates how P2P routing performance is improved in semantic space.

4.1 An Example of Query Routing in Semantic Space

A node can semantically link itself to neighbors if it knows semantic descriptions of the resources they own. Fig. 8 shows a partial semantic link network, where B, C, D, and E are keyword sets describing the resources owned by corresponding nodes. A new node with keyword set X can establish semantic links with neighbors by comparing keyword sets. Assume $X \supseteq B$ (that is, any keyword in B is also in X), $X \subseteq D$, and X is irrelevant to C (that is, $X \cap C = \{\}$, null link exists between X and C).

When receiving a query Q (a keyword set describing query content), X compares itself with Q first. If $Q \subseteq X$, then the answer can be found in X and Q should not be forwarded to C since Q and C have no common content (this is implied by $Q \subseteq X$ and $X \cap C = \{\}$). If $Q \cap X = \{\}$, then Q should not be forwarded to B as X covers B. If $Q \supset X$, then Q should not be forwarded to B since B does not contain more answers than X, Q should be forwarded to D or C since D may contain more answers than B ($B \subseteq X \subseteq D$), and C may contain answers different from X ($X \cap C = \{\}$).

Differently from an ordinary network, the semantic link network could imply some semantic links. For example, E becomes the neighbor of X since $X \supseteq E$ can be derived from $X \supseteq B$ and $B \supseteq E$. This implied semantic link influences messages flowing through the link from B to E as some messages flow directly from X to E when the new link is established.

This example illustrates that semantic links can effectively navigate queries in P2P networks. Different semantic description approaches like Resource Description Framework Schema (RDFS) [9], Fuzzy Cognitive Map (Fuzzy CM) [20], and Web Ontology Language (OWL, www.w3.org/2004/OWL/) can be also used to reflect the nodes' semantics. The corresponding mechanisms for determining the semantic relations between nodes are needed.

4.2 Semantic Link Network

The semantic link is the extension of the hyperlink in semantics for describing various semantic relations between resources. A semantic link X—α→ Y represents semantic relation α between semantic nodes X and Y. α can be *ins* (*instance-of*), *subtype, similar-to, cause-effect, implication, part-of, attribute-of, sequential, reference, equal-to, unknown, feature-of*, and *opposite*. For example, the *subtype* relationship between concepts constitutes concept networks, where a *subtype* concept inherits all features and relations of its superconcept and can include more features. A semantic relation can be instantiated according to applications, for example, the *reference* relation can be instantiated as the citation relation between papers and the foreign key relation between relational tables.

A semantic link network consists of semantic nodes and semantic links. A semantic node can be an entity, a concept, a schema (a structure on features), or a semantic community. An entity is a point in the feature space: $\{feature : value; \ldots ; feature : value\}$. A concept is described by a *feature space* and a set of semantic relations on features L (can be null): $< \{feature : type; \ldots ; feature : type\}, L >$. A semantic community is a semantic link network that represents integrated semantics and has no isolated nodes or parts. A semantic link network usually consists of a conceptual layer and an entity layer, as introduced in [43].

The semantic link network has two major characteristics: 1) Potential semantic links could be derived according to existing semantic links and certain semantic link rules like X—α→ Y, Y—β→ Z ⇒ X—γ→ Z (denoted as $\alpha \cdot \beta \Rightarrow \gamma$ for constructing semantic link calculus) [39]. 2) It integrates algebra representation, algebra calculus, logical reasoning, and analogical reasoning [43].

A semantic space can be defined by a five-tuple:

$$< feature\text{-}space, concept\text{-}space, axiom\text{-}space,\\ semantic\text{-}link\text{-}space, semantic\text{-}operations > .$$

Semantic operations include set union ($\cup$), intersection ($\cap$), and product ($\times$). The $concept\text{-}space$ and $axiom\text{-}space$ are sets of concepts and axioms and

$$\begin{aligned} semantic\text{-}link\text{-}space = {} & concept\text{-}space \times concept\text{-}space \\ & \cup\ axiom\text{-}space \times axiom\text{-}space \\ & \cup\ feature\text{-}space \times feature\text{-}space. \end{aligned}$$

A semantic link network can be regarded as the result of mapping nodes of ordinary network into the feature space, concept space, and axiom space, and mapping links into the semantic link space.

Previous researches on semantic P2P mainly focus on the use of similarity between peers to improve the performance of P2P network [13], [21], [24], [35]. Their researches also provide evidences for our assertion: The performance of P2P can be improved in semantic space as the similarity between nodes is actually the *similar-to* semantic link.

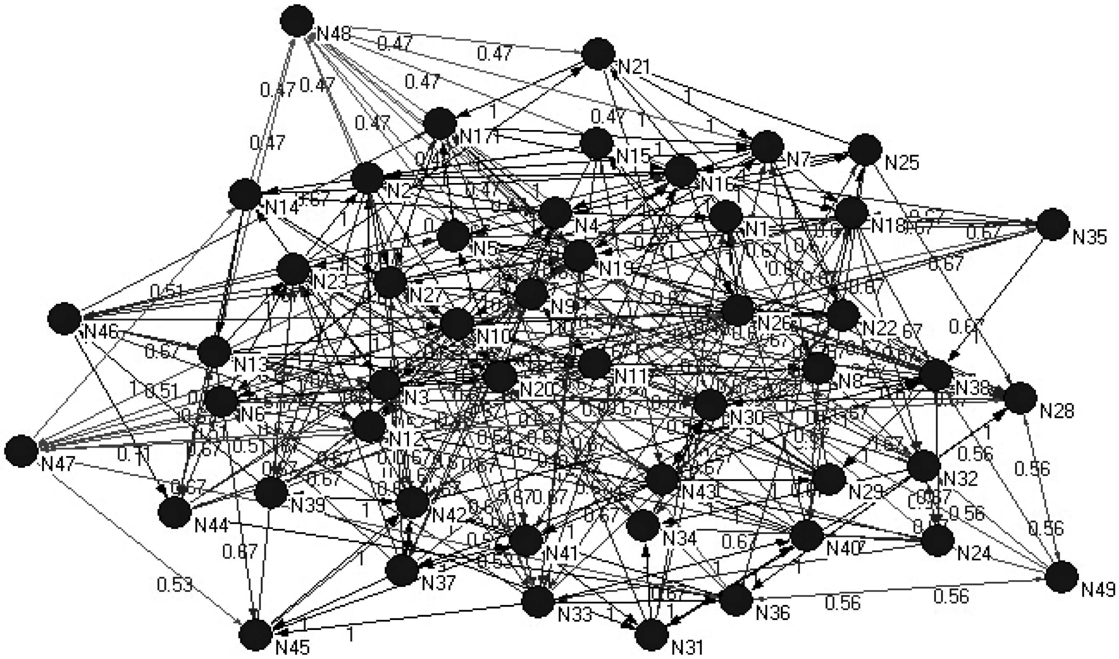

Fig. 9. A real P2P semantic link network of 49 nodes.

4.3 Query Routing in Semantic Link Network: An Experiment

Here, we investigate the efficiency of routing a query in a real semantic link network of 49 nodes. Each node uses schema to describe interested information. The metadata of 500,000 papers is collected from DataBase systems and Logic Programming (DBLP) XML databases (http://dblp.uni-trier.de/xml) and ACM SIGMOD XML records (www.acm.org/sigmod/record/xml/SigmodRecord/SigmodRecord.xml) and then the metadata is distributed onto 49 nodes according to the nodes' schemas. The experiment assumes that each node has the same bandwidth and processing ability. Three hundred links are added between nodes, so, on the average, each node randomly selects 6.12 neighbors.

With reference to the methods introduced in [24] and [41], the *similar-to* links are obtained and used to connect nodes to form a P2P semantic link network, as shown in Fig. 9, where the values on the links represent the similarities between the nodes' schemas and darker links represent larger similarity. Upon receiving a query, a node forwards it with higher probability to neighbors that have more similar schemas.

Twenty randomly selected nodes emit 20 keyword-based queries on this semantic link network. Each query is forwarded by using the following routing methods:

1. *Flooding mechanism (denoted as flooding)*. Each node forwards the query to all its neighbors except the one it receives the query from.
2. *Flat gossip mechanisms (denoted as gossip)*. Each node forwards the query to $fanout$ number of randomly selected neighbors.
3. *Semantic-based gossip mechanisms (denoted as SGM in* Fig. 10). Each node forwards the query to a $fanout$ number of neighbors according to the semantic similarities between nodes.

The performance metric used here is the ratio of recall to the number of messages sent for the queries (*recall*/*message*). The results are averaged for generality. The experimental results in Fig. 10 show the following phenomena:

1. The semantic-based mechanisms achieve the best performance compared with the gossip mechanisms, with $fanout$ ranging from 1 to 4 and TTL ranging from 1 to 12 or compared with the flooding mechanism with TTL ranging from 1 to 6.
2. The gossip mechanisms show irregular results due to its random characteristic when selecting neighbors.
3. The gossip mechanisms perform better than the flooding mechanism most of the time. The flooding mechanism achieves better performance than the gossip only when TTL is small and, when TTL gets larger, its performance decreases quickly since it almost reaches all the nodes in the network when TTL is 4 in this experiment. In this situation, the continually increasing TTL value will produce

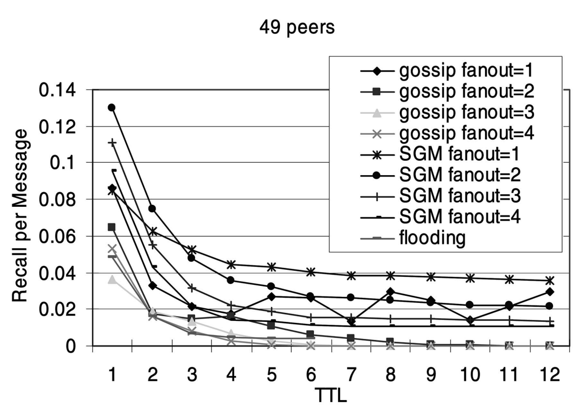

Fig. 10. Comparisons among gossip, semantic-based, and flooding mechanisms.

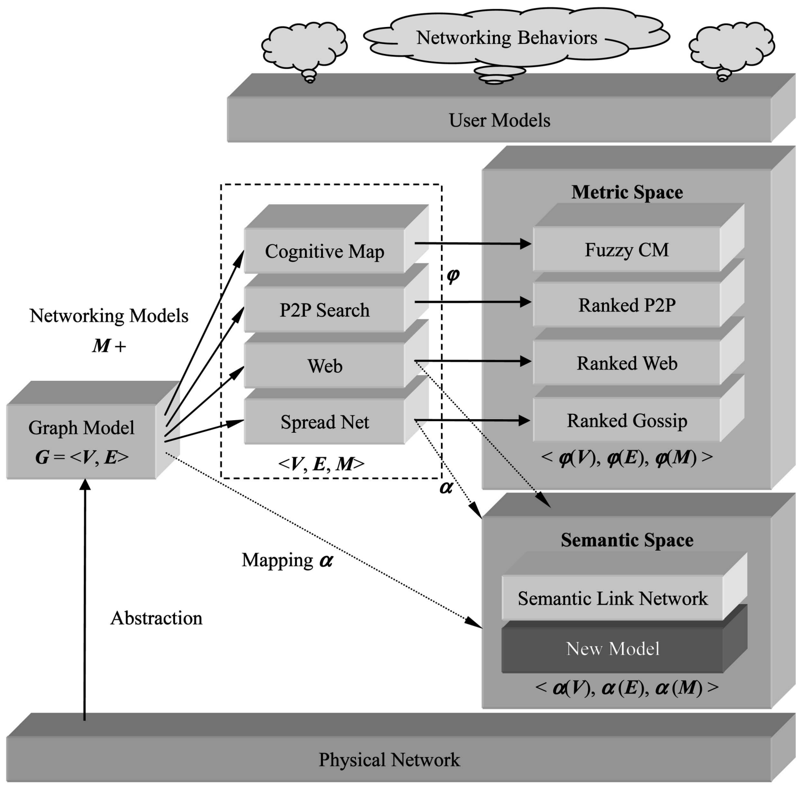

Fig. 11. A general framework of networking with metric space and semantic space.

redundant messages transmitted on the network and, consequently, reduce the performance metric: *recall/message*.

Gossip in a semantic link network works differently than in an ordinary network since some information is only disseminated via some semantic links. A general gossip mechanism on semantic link network includes the following steps:

1. Each node compares the receiving message with its own content, and then selects neighbors semantically relevant to the gossip message. A solution for doing this is to obtain a set of keywords of the gossip message and compare them with the nodes' feature set. It is reasonable to assume that the more the features of a node are relevant to the keywords, the more the node is interested in the message.
2. Determine the semantic links relevant to the gossip message.
3. Obtain a semantic view of the local semantic link network about the gossip message. A semantic view about this gossip message consists of semantically relevant nodes and the semantic links between them.
4. Map the view into the metric space to obtain the ranks of semantic links.
5. Carry out the semantic gossip in the ranked semantic link network.

5 Navigation in Metric Space and Semantic Space

A type of network can be generalized as a networking model M on node set V and edge set E, denoted as $< V, E, M >$. Different applications use the network with different networking models [40].

The networking model of the World Wide Web is a self-organized hyperlink network. A page can link to any page in terms of URL. Search engines select, index, and retrieve URLs according to keywords. Page rank was proposed to differentiate the importance among Web pages. The rank of a page is determined by the number of incoming links and the ranks of nodes pointing to it. Various ranking strategies have been developed to improve search efficiency [17]. Research and practice have shown the reasonability of the rank-aware search engines [8], [14]. Human activities on the Web and the self-organization networking model of the Web lead to the scale-free and power-law phenomena [5], which accord with the assumption that pages with higher ranks are more useful to users.

Essentially, the accuracy of information retrieval can be improved by mapping a network into a metric space to differentiate nodes and links. The underlying assumption of this type of approach is that a shorter distance between nodes in the metric space implies closer contents.

Although effects could be raised by improving the mapping mechanisms and the networking model, the metric space does not reflect semantics. For example, the content of two Web pages could be different even though they have the same page rank. The semantic space can differentiate nodes (or links) in semantics.

The Semantic Web can be regarded as an attempt to establish a mapping from the World Wide Web into the semantic space featured by ontologies and markup languages [6], [9].

Fig. 11 shows a general framework of networking with the metric space and the semantic space. The mapping mechanism φ maps network $< V, E, M >$ into the metric space to get a ranked network $< \varphi(V), \varphi(E), \varphi(M) >$. The mapping mechanism α maps network $< V, E, M >$ into the semantic space to get a semantic link network $< \alpha(V), \alpha(E), \alpha(M) >$. New models can be obtained by mappings from a given network into the metric space or the semantic space. For example, the Fuzzy CM is the result of mapping the cognitive map into the (fuzzy) metric space [20]. The ranked gossip is the result of mapping the spread net into the metric space. A different type of networking model M may require different mapping φ. Moreover, the algorithm of $\varphi(M)$ could be different from that of M.

The high-level models like social and economic models reflect the rules of using a network [35]. The user models help organize information according to users' behaviors. For example, users who are interested in the same category of information could be organized in the same community. Such localization could also raise the efficiency of information service.

This framework indicates that *given a new type of network, the efficiency of using the network can be raised by appropriately mapping it into metric space or semantic space*. The efficiency can be further raised if the application-level models and the user models are incorporated. This is useful for research on effective use of a new network or the development of a large-scale network.

6 Conclusion

Three improved gossip mechanisms are first proposed by mapping a network into metric space and dynamically adapting the number of selected neighbors to disseminate messages. By 100 times comparison experiment with two typical networks of 1,000 nodes, two metrics show that the mapping mechanisms perform better than ordinary gossip mechanisms. Further investigation shows that routing in a P2P semantic link network can also obtain higher performance than routing in metric space. This study and analysis of relevant works confirm our conjecture that the performance of P2P information service can be improved by appropriately mapping a network into metric space or semantic space and establishing the corresponding information control mechanisms. The suggested general framework for networking with metric space and semantic space helps research and development of an efficient network.

Acknowledgments

The authors would like to thank the reviewers, who provided constructive comments on the earlier version of this paper. They also would like to thank all of the team members of the China Knowledge Grid Research Group for their help and cooperation. The research work is supported by the National Basic Research Program of China (973 Grant No. 2003CB317001), the International Cooperation Project of Ministry of Science and Technology of China (Grant No. 2006DFA11970), the EU Sixth Framework Program GREDIA (IST-FP6-034363), and the National Science Foundation of China.

References

[1] K. Aberer et al., "Indexing Data-Oriented Overlay Networks," *Proc. 31st Int'l Conf. Very Large Data Bases,* pp. 685-696, 2005.

[2] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies," *ACM Computing Surveys,* vol. 36, no. 4, pp. 335-371, 2004.

[3] N.T.J. Bailey, *The Mathematics Theory of Infectious Diseases and Its Applications,* second ed. Hafner Press, 1975.

[4] H. Balakrishnan et al., "Looking Up Data in P2P Systems," *Comm. ACM,* vol. 46, no. 2, pp. 43-48, 2003.

[5] A.L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science,* vol. 286, pp. 509-512, 1999.

[6] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific Am.,* vol. 284, no. 5, pp. 24-43, 2001.

[7] K.P. Birman et al., "Bimodal Multicast," *ACM Trans. Computer Systems,* vol. 17, no. 2, pp. 41-88, 1999.

[8] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *Proc. Seventh Int'l World Wide Web Conf.,* pp. 107-117, 1998.

[9] J. Broekstra, M.C.A. Klein, S. Decker, D. Fensel, F.V. Harmelen, and I. Horrocks, "Enabling Knowledge Representation on the Web by Extending RDF Schema," *Proc. 10th Int'l World Wide Web Conf.,* pp. 467-478, 2001.

[10] A. Demers et al., "Epidemic Algorithms for Replicated Database Maintenance," *Proc. Sixth Ann. Symp. Principles of Distributed Computing,* pp. 1-12, 1987.

[11] P.T. Eugster et al., "Epidemic Information Dissemination in Distributed Systems," *Computer,* pp. 60-67, 2004.

[12] A.J. Ganesh, A.M. Kermarrec, and L. Massoulié, "Peer-to-Peer Membership Management for Gossip-Based Protocols," *IEEE Trans. Computers,* vol. 52, no. 2, pp. 139-149, Feb. 2003.

[13] P. Haase et al., "Bibster: A Semantics-Based Bibliographic Peer-to-Peer System," *Proc. Int'l Semantic Web Conf.,* pp. 122-136, 2004.

[14] T.H. Haveliwala, "Topic-Sensitive Pagerank: A Context-Sensitive Ranking Algorithm for Web Search," *IEEE Trans. Knowledge and Data Eng.,* vol. 15, no. 4, pp. 784-796, July/Aug. 2003.

[15] W.O. Kermack and A.G. Mckendrick, "Contributions to the Mathematical Theory of Epidemics," *Proc. Royal Soc. London,* vol. 115, pp. 700-721, 1927.

[16] A.M. Kermarrec, L. Massoulié, and A.J. Ganesh, "Probabilistic Reliable Dissemination in Large-Scale Systems," *IEEE Trans. Parallel and Distributed Systems,* vol. 14, no. 3, pp. 248-258, Mar. 2003.

[17] J. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *Proc. Ninth Ann. ACM-SIAM Symp. Discrete Algorithms,* pp. 668-677, 1998.

[18] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson, "On the Self-Similar Nature of Ethernet Traffic," *IEEE/ACM Trans. Networking,* pp. 1-15, 1994.

[19] M.J. Lin, K. Marzullo, and S. Masini, "Gossip Versus Deterministically Constrained Flooding on Small Networks," *Proc. 14th Int'l Conf. Distributed Computing,* pp. 253-267, 2000.

[20] Z.Q. Liu and R. Satur, "Contextual Fuzzy Cognitive Maps for Decision Support in Geographic Information Systems," *IEEE Trans. Fuzzy Systems,* vol. 7, no. 10, pp. 495-502, 1999.

[21] A. Löser, S. Staab, and C. Tempich, "Semantic Methods for P2P Query Routing," *Proc. German Conf. Multi Agent Technologies (MATES '05),* pp. 15-26, 2005.

[22] Q. Lv et al., "Search and Replication in Unstructured Peer-to-Peer Networks," *Proc. ACM Int'l Conf. Supercomputing,* pp. 84-95, 2002.

[23] A. Medina, I. Matta, and J. Byers, "On the Origin of Power Laws in Internet Topologies," *Computer Comm. Rev.,* vol. 30, no. 2, pp. 18-28, 2000.

[24] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching," *Proc. 18th Int'l Conf. Data Eng.,* pp. 117-128, 2002.

[25] V. Paxson and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Trans. Networking,* vol. 3, pp. 226-244, 1995.

[26] B. Pittel, "On Spreading a Rumor," *SIAM J. Applied Math.,* vol. 47, no. 1, pp. 213-223, 1987.

[27] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks," *Proc. 35th Ann. Hawaii Int'l Conf. System Sciences (HICSS '02),* vol. 9, pp. 298-307, 2002.

[28] S. Ratnasamy et al., "A Scalable Content-Addressable Network," *Proc. ACM SIGCOMM Symp. Comm., Architecture, and Protocol,* pp. 161-172, 2001.

[29] R.V. Renesse, K.P. Birman, and W. Vogels, "Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining," *ACM Trans. Computer Systems,* vol. 21, no. 2, pp. 164-206, 2003.

[30] R.V. Renesse, Y. Minsky, and M. Hayden, "A Gossip-Style Failure Detection Service," *Proc. Int'l Federation for Information Processing (IFIP) Int'l Conf. Distributed Systems and Platforms and Open Distributed Processing (Middleware '98),* pp. 55-70, 1998.

[31] M. Ripeanu, I. Foster, and A. Iamnichi, "Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design," *IEEE Internet Computing,* vol. 6, no. 1, pp. 50-57, 2002.

[32] A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems," *Proc. Int'l Conf. Distributed Systems Platforms,* pp. 329-350, 2001.

[33] N. Sarshar, P.O. Boykin, and V.P. Roychowdhury, "Percolation Search in Power Law Networks: Making Unstructured Peer-to-Peer Networks Scalable," *Proc. Fourth Int'l Conf. Peer-to-Peer Computing,* pp. 2-9, 2004.

[34] R.M. Stoica et al., "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *Proc. ACM Conf. Applications, Technologies, Architectures, and Protocols for Computer Comm. (SIGCOMM '01),* 2001.

[35] C. Tempich, S. Staab, and A. Wranik, "Reminding: Semantic Query Routing in Peer-to-Peer Networks Based on Social Metaphors," *Proc. 13th Int'l World Wide Web Conf.,* pp. 640-649, 2004.

[36] W. Vogels, R.V. Renesse, and K. Birman, "The Power of Epidemics: Robust Communication for Large-Scale Distributed Systems," *ACM SIGCOMM Computer Comm. Rev.,* vol. 33, no. 1, pp. 131-135, 2003.

[37] S. Voulgaris, M. Jelasity, and M.V. Steen, "A Robust and Scalable Peer-to-Peer Gossiping Protocol," *Peer-to-Peer Computing,* Springer, 2003.

[38] B.Y. Zhao, J.D. Kubiatowicz, and A.D. Joseph, "Tapestry: An Infrastructure for Fault-Resilient Wide-Area Location and Routing," Technical Report UCB/CSD-01-1141, Univ. of California, Berkeley, 2001.

[39] H. Zhuge, *The Knowledge Grid.* World Scientific, 2004.

[40] H. Zhuge, "Exploring an Epidemic in an E-Science Environment," *Comm. ACM,* vol. 48, no. 9, pp. 109-114, 2005.

[41] H. Zhuge, J. Liu, L. Feng, X. Sun, and C. He, "Query Routing in a Peer-to-Peer Semantic Link Network," *Computational Intelligence,* vol. 21, no. 2, pp. 197-216, 2005.

[42] H. Zhuge, X. Sun, J. Liu, E. Yao, and X. Chen, "A Scalable P2P Platform for the Knowledge Grid," *IEEE Trans. Knowledge and Data Eng.,* vol. 17, no. 12, pp. 1721-1736, Dec. 2005.

[43] H. Zhuge, "Autonomous Semantic Link Networking Model for the Knowledge Grid," *Concurrency and Computation: Practice and Experience,* DOI: 10.1002/cpe.1097, 2006.

Hai Zhuge is the chief scientist of the China Semantic Grid project funded by the National Basic Research Program of China. He is a professor and the director of the Key Lab of Intelligent Information Processing in the Institute of Computing Technology at the Chinese Academy of Sciences and the founder of the China Knowledge Grid Research Group (http://kg.ict.ac.cn). He was a keynote speaker at many international conferences like the Web-Age Information Management Conference (WAIM), the Asian Semantic Web Conference (ASWC), the International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), and the International Conference on Electronic Commerce (ICEC). He organizes the International Conference on Semantics, Knowledge, and Grid (www.knowledgegrid.net). He serves as the area editor of the *Journal of Systems and Software* and the associate editor of *Future Generation Computer Systems*. His monograph *The Knowledge Grid* is the first book in the area. His publications also appeared in leading periodicals such as *Communications of the ACM*, *Computer*, and the *IEEE Transactions on Knowledge and Data Engineering*. Some were in the top 1 percent of highly cited papers according to the Institute for Scientific Information (ISI) Essential Science Indicator. He was the top scholar in the systems and software engineering area (1999-2004) according to the assessment report of the *Journal of Systems and Software*. He is a senior member of the IEEE and a member of the ACM.

Xiang Li is a PhD student and a research fellow of the China Knowledge Grid Research Group in the Institute of Computing Technology at the Chinese Academy of Sciences. His research interests include peer-to-peer computing, knowledge grid, and future networking techniques.