J Supercomput (2009) 48: 264–285
DOI 10.1007/s11227-008-0219-8

A structured P2P network based on the small world phenomenon

Jie Xu · Hai Jin

Published online: 5 July 2008

Abstract In this paper, we propose a new structured P2P overlay network, named SW-Uinta(small-world). In order to reduce the routing latency, we firstly construct the Uinta network in which both physical characteristics of network and data semantic are considered. Furthermore, based on Uinta, a nondeterministic caching strategy is employed to allow for poly-logarithmic search time while having only a constant cache size. Compared with the deterministic caching strategy proposed by previous P2P systems, the nondeterministic caching strategy can reduce communication overhead for maintaining the routing cache table. Cache entries in the cache table of peer nodes can be updated by subsequent queries rather than only by running stabilization periodically. In the following, a novel cache replacement scheme, named the SW cache replacement scheme, is used to improve lookup performance, which has proved to satisfy the small-world principle. So we call this network SW-Uinta(small-world). After that, according to the theoretical analysis, it can be proved that SW-Uinta(small-world) can get $O((\log^2 N)/k)$ search time with $O(k)$ cache size. Lastly, the performance of SW-Uinta(small-world) is compared with those of other structured P2P networks such as Chord and Uinta. It shows that SW-Uinta(small-world) can achieve improved object lookup performance and reduce maintenance cost.

This paper is supported by National Science Foundation of China under Grant 60433040, and CNGI projects under Grants CNGI-04-12-2A and CNGI-04-12-1D.

J. Xu (✉) · H. Jin
Services Computing Technology and System Lab., Cluster and Grid Computing Lab., School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
e-mail: jiexu@hust.edu.cn

H. Jin (✉)
e-mail: hjin@hust.edu.cn

J. Xu
Faculty of Mathematics and Computer Science, Hubei University, Wuhan 430062, China

Keywords Peer-to-peer · Physical topology · Small world phenomenon · Cache replacement scheme

1 Introduction

P2P systems are self-organizing distributed systems with no centralized control. Each peer in the P2P network has similar functionalities and plays the roles of a server and a client at the same time. These systems have recently gained much attention, primarily because of the great number of features they can offer to applications that are built on top of them, such as scalability, availability, fault tolerance, decentralized administration, anonymity, and so on.

Generally, current P2P networks can be classified into two types, namely unstructured and structured. Unstructured systems like Gnutella [1], Kazaa [2], and Freenet [3] are constructed without any regularization on the connectivity among peers and the routing mechanism. Such networks are easier to build and maintain (or the lack of it). Typically, new peers randomly connect to existing alive nodes in the network and the search process for data is flooding (called as Breadth First Search) or iterative searching (called as Depth First Search). However, the flooding-based searching mechanism consumes too much bandwidth to be suitable for large systems. Moreover, for them, the emphases are on fast file retrieval, with no guarantee that files will always be located.

In contrast, structured P2P systems such as Chord [4], CAN [5], Pastry [6], and Tapestry [7] follow a predetermined structure. They typically use Distributed Hash Table (DHT) functionality to assign identifiers to nodes and data. These systems are constructed in overlay networks at the application layer without taking the physical network topologies into consideration, which can result in high lookup delays and unnecessary wide-area network traffic when a routing hop takes a message to a peer with a random location on the Internet. However, they guarantee that the file will always be located at the cost of increased overhead for peers joining/leaving and the routing table maintaining.

In order to reduce lookup delays, some researchers have proposed several DHT-based network infrastructures using physical topology information [8], which map the overlay's logical identifier space onto the physical network such that neighboring nodes in the logical space are close in the physical network. But all of these systems ignore users' interests and do not consider data semantic.

To address both users' interests and the physical topology, we firstly propose an overlay network named Uinta. All of peers are grouped into several clusters based on the physical topology of network, which makes peers in the same cluster have small link latency and peers in the different cluster have long link latency. Because users always tend to retrieve data of a kind which they are interested in, data information is stored based on data semantics, which makes data of a kind place in the same cluster. Of course, how to analyze data semantic and how to decide the data class can be decided by the mature technology in the field of data mining, which is out of scope of this paper. After that, the user can utilize a class cache table to cache the identifier of peer where data of some kind searched recently store and the identifier of this

kind. If the user searches data of this kind next time, it can use the information of the cache table directly. It is obvious that P2P system workload has temporal and spatial localities just as that in the web traffic [9]. A high class cache table hit rate can be expected, thus a reduced average number of routing hops and lower routing network latency can be achieved.

In addition, most of the existing structured P2P systems adopt the deterministic caching scheme. The deterministic caching scheme means that which keys should be addressed in the cache of peer N is based on the key of peer N and cached index entries typically have expiration times after which they are considered stale. However, little attention has been given to how to maintain these caches during the lookup process. Therefore, communication overhead will be high for maintaining the routing cache table.

In order to improve the performance and reduce maintenance cost further, we propose the SW-Uinta overlay network based on Uinta. In SW-Uinta, a nondeterministic caching scheme is used to maintain routing cache tables. The nondeterministic caching scheme updates cache index entries after answering search queries which can reduce maintenance cost. The basic idea is to arrange all peers along a ring-over-ring and equip them with some short distance contacts and long distance contacts. Short distance contacts are built when the peer joins the system with its immediate neighbors. Long distance contacts are built after the peer receives the reply that it requests. Assume that peer S initiates the query for key K which is in the cache of peer T. Upon receiving the answer from peer T, peer S caches the information of peer T which arrives with the reply. And the traditional cache replacement scheme such as LRU, LFU, and FIFO etc. cannot result in the better lookup performance because they do not consider the network topology. In order to optimize the performance of global system, we construct SW-Uinta(small-world) to use the intuition from the small world model [10–14] which says that the routing distance in a graph will be small if each peer has pointers pointing to its immediate neighbors and some chosen far away nodes.

Compared with Uinta, extensional aims of the SW-Uinta(small-world) system are:

(1) A nondeterministic caching scheme is proposed to reduce maintenance cost for updating the routing cache table.
(2) The SW cache replacement scheme with the small-world paradigm is proposed to further improve the performance of object lookup.

The rest of this paper is organized as follows. In Sect. 2, an overview of related works is presented. In Sect. 3, we provide the method how to construct the Uinta overlay network and SW-Uinta(small-world) overlay network. Several protocols for the SW-Uinta(small-world) are proposed in Sect. 4. In Sect. 5, theoretical analysis for Uinta and SW-Uinta(small-world) are described. Simulations are discussed and results show the performance of SW-Uinta(small-world) outperforms some of those other systems in Sect. 6. We conclude our research and propose future work in the last section.

2 Related works

In this section, features of related P2P networks are described.

Chord [4]. It is based on a ring topology structure. Each peer is mapped onto a 1-dimensional circular coordinate space from 0 to $2^m - 1$. The peer who is responsible for a certain *key* is the first peer to succeed the *key*. Each peer maintains three sets of neighbors' information: a predecessor, a successor list, and the finger table of $O(\log N)$ peers spaced exponentially around the key space. To locate a key, every peer simply routes the message to the peer which is the biggest, but does not overshoot the key in its routing table until the key lies between the peer and its successor, which means that the successor is the destination peer. Each peer has $O(\log N)$ neighbors. The routing path is $O(\log N)$ hops and maintenance cost for updating the routing table is $O(\log^2 N)$.

CAN [5]. It has a hypercube topology structure. Each peer is mapped onto a d dimensional coordinate space and attached to a hypercube region in this space. Each peer corresponds to one zone and stores the data that are mapped to this zone by hash function. In d-dimensional space, two peers are neighbors if their corresponding zones overlap along $d-1$ dimension, but are neighbors to each other in the remaining dimension. Each peer in CAN maintains only information about its immediate neighbors. To locate a key, a query peer simply routes the message to the neighbors which makes the most progress to the destination until the destination peer is reached. Each peer maintains $O(d)$ neighbors; the routing path is $O(dN^{1/d})$ hops and maintenance cost for updating the routing table is $O(d)$.

Pastry [6]. Its underlying overlay network is of tree alike topology structure. Each peer maintains three sets of neighbors' information: leaf set, neighborhood set, and routing table. To locate a key, the peer first checks to see if the key falls in the range of its leaf set. If so, the message is forwarded directly to the destination peer. If the key is not covered by the leaf set, the peer routes the message to the peer whose *ID* shares the longest prefix with the key in its routing table. In Pastry, each peer has $O(B\log_B N)$ neighbors; routing path is $O(\log_B N)$ hops($B = 2^b$ and b is a configuration parameter with typical value 4) and the maintenance cost for updating the routing table is $O(\log_B^2 N)$.

Tapestry [7]. Analogous to Pastry, it also has tree-alike topology structure. Each peer maintains a neighbor map which is composed of $\log_B N$ levels and a back pointer list that points to peers where it is referred to as a neighbor. Each level in the neighbor map represents a matching suffix up to a digit position in the *ID*. Every levels of the neighbor map contains B entries. To locate a key, each peer routes the message to the neighbor whose *ID* shares the longest suffix with the key in its neighbor map. Each peer has $O(B\log_B N)$ neighbors; the routing path is $O(\log_B N)$ hops and maintenance cost for updating the routing table is $O(\log_B^2 N)$, in a system with N peers using NodeIDs of base B.

Kademlia [15]. Like Chord, it has a ring alike topology structure. Each peer maintains $O(\log N)k$-buckets and each k-bucket is kept sorted by time last seen, i.e., least recently seen node at the head, most recently seen at the tail. And the Kademlia system uses the numerical of the Exclusive OR(XOR) as the routing distance between two nodes. To locate a key, a peer starts by performing a FIND-VALUE lookup to

find the k peers with IDs closest to the key. Each peer has $O(k \log N)$ neighbors; the routing path is $O(\log N)$ hops and maintenance cost for updating the routing table is $O(\log^2 N)$.

Viceroy [16]. Its underlying overlay network is of butterfly alike topology structure and it is the first proposal that provides $O(\log N)$ routing latency with only a constant number of links. Like Chord, nodes are placed along a circle. A node additionally belongs to one out of approximately $O(\log N)$ concentric rings lying one above the other. These rings correspond to layers in Butterfly networks. A node maintains connections with two neighbors each along the two rings in which it belongs. It also maintains two connections to a pair of nodes in a lower ring and one connection with a node in the ring above. Routing requires $O(\log N)$ hops on average and maintenance cost for updating the routing table is $O(\log^2 N)$.

All of the above P2P systems are constructed without taking the small world model into account.

Symphony [17]. It is also a kind of system that extends Chord network structure by additional direct links. The core idea of Symphony is to place all peers along a ring and attach each peer with a few long distance links. Symphony is inspired by Kleinberg's Small World construction. It is shown that with $k = O(1)$ links per node and it is possible to route hash lookups with an average latency of $O((\log^2 N)/k)$ hops and maintenance cost for updating the routing table is $O(\log^2 N)$. In Symphony, the small-world phenomenon is considered. However, little attention has been given to how to maintain intermediate caches. Peers in these systems cannot dynamically change their long links so that reestablishment of all long links is a complex process.

CUP [18] is a protocol for performing Controlled Update Propagation to build and to maintain caches of index entries in peer-to-peer networks. The experiments have shown that CUP significantly reduces average query latency and any propagation overhead incurred by CUP is compensated for by a factor of 2 to 200 times in terms of savings in cache misses. However, they have not considered the small world phenomenon in the P2P network.

Some P2P systems are also constructed with taking the topology mismatch problem into account.

TOPLUS [19] is a novel lookup service for structured peer-to-peer networks that is based on the hierarchical grouping of peers according to network IP prefixes, gathered from such sources as BGP tables and repositories of well-known prefixes. However, the scheme of clustering peer nodes based on network IP prefixes is not suitable because network IP prefixes could not show geographic properties of peers exactly. Peer nodes within contiguous prefixes could not be adjacent nodes.

SkipNET [20] is a scalable overlay network that provides controlled data placement and guaranteed routing locality by organizing data primarily by string names. SkipNet clusters nodes according to their name ID ordering, nodes within a single organization gracefully survive failures that disconnect the organization from the rest of the Internet. However, the problem of this clustering scheme is similar to that for TOPLUS, that is, peer nodes within the same level could be distant nodes.

Coral [21] is a peer-to-peer content distribution system. Coral creates self-organizing clusters of nodes that fetch information from each other to avoid communicating with more distant or heavily-loaded servers. Coral indexes data, but does

not store it. The actual content resides where it is used, such as in nodes' local web caches. In the case of clustering nodes, Coral has the similar idea to our scheme.

So when constructing the Uinta, both the physical topology of network and data semantic are considered. Except for these, when constructing the SW-Uinta(small-world), how to reduce maintenance cost and how to improve the performance with the small world model are proposed.

3 SW-Uinta(small-world) overlay network

In this section, we firstly show how to incorporate the underlying topological information and the data semantic in the construction of the Uinta overlay to improve the routing performance. After that, we present protocols for constructing and maintaining a small world P2P network SW-Uinta(small-world).

3.1 Construction of the Uinta overlay network

Uinta is a two-layer structured overlay in which peers are organized in different clusters, and routing messages are first routed to the destination cluster using an intercluster overlay, and then routed to the destination peer using an intragroup overlay. Our scheme is built on the structured system which can guarantee that each file can be located. We take a torus overlay structure in the Chord system to construct Uinta for both of layers because the ring geometry allows the greatest flexibility, and hence achieves the best resilience and proximity performance [22].

Construction of Uinta overlay network involves three major tasks: (1) forming peer clusters based on the topology of physical network; (2) assigning an identifier to a peer or a key that positions it in the peer group; and (3) construct an overlay network across the peer clusters.

Cluster formation The goal of our clustering scheme is to have a set of peers independently partition themselves into disjoint clusters such that peers within a single cluster are relatively closer to one another than to peers not in their cluster. So the peers should be organized into clusters based on the topology of physical network. And the cluster formation strategy has great impact on Uinta efficiency. It must be relatively simple and fast with minimal overhead since peers may join/leave system frequently. Also, it must be approximately accurate and will group the relatively close peers into the same cluster. Otherwise, the average link latencies have no big difference in the different cluster and the same cluster.

A simple and relatively accurate topology measurement mechanism is the distributed binning scheme proposed by Ratnasamy and Shenker [8]. In this scheme, a well-known set of machines are chosen as the landmark nodes. System peers partition themselves into disjoint bins such that peers that fall within a given bin are relatively closer to each other in terms of network link latency. Although the network latency measurement method (*ping*) is not very accurate and determined by many uncertain factors, it is adequate for Uinta and we use the method which is similar with that in [8] for cluster formation.

Table 1 Sample peers in a Uinta system with three landmark nodes

Peer	Dist-L1	Dist-L2	Dist-L3	Cluster name
A	110 ms	150 ms	240 ms	112
B	22 ms	135 ms	235 ms	012
C	285 ms	264 ms	45 ms	220
D	260 ms	244 ms	67 ms	220
E	30 ms	120 ms	220 ms	012
F	28 ms	115 ms	225 ms	012

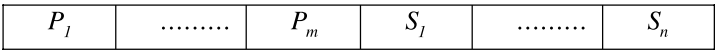


Fig. 1 The binary format of identifier in the Uinta system

Table 1 shows 6 sample nodes A, B, C, D, E, and F in a Uinta system with the measured network link latencies to 3 landmark nodes L1, L2, and L3. We may divide the range of possible latency values into 3 levels; level 0 for latencies in the range [0, 100) ms, level 1 for latencies between [100, 200) ms, and level 2 for latencies no smaller than 200 ms. The cluster name is created according to the measured latencies to the 3 landmark nodes, and this information is used for peers clustering. For example, Peer A's landmark information is 112. Peer C and peer D have the same information 220 so that they are in the same cluster named 220. And all the other nodes are belong to cluster named 012.

Assignment of identifier This step also includes three tasks: assignment of peer id, cluster id, and key id. In Uinta, each cluster and each key is composed of two parts: m-bit prefix and n-bit suffix. To peer id, m-bit prefix is assigned to the identifier of a cluster that the peer belongs to and n-bit suffix is assigned to the identifier chosen by hashing the peer's IP address. To key id, m-bit prefix is assigned as the identifier of a class that the key belongs to and n-bit suffix is assigned to the identifier chosen by hashing the key. To cluster id, m-bit prefix is assigned to the identifier generated by hashing the cluster name and n-bit suffix is assigned to 0. The consistent hash function such as SHA-1 [23] is used to avoid the possible identifier duplication problem.

The binary format of identifier in the Uinta system is shown in Fig. 1, in which P_i and S_j $(i = 0, 1 \ldots, m$ and $j = 0, 1, \ldots, n)$ are assigned to 0 or 1. $P_1, \ldots, P_m$ is denoted by the prefix of an identifier that is marked as P, which is the identifier of cluster or class and $S_1, \ldots, S_n$ is denoted by the suffix of an identifier that is marked as S, which is hashed by the peer's IP address or the key, so that the identifier is equal to $D = P \cdot 2^n + S$.

Uinta overlay network construction To construct the overlay, each peer p in the Uinta system maintains two finger tables: c-finger table and l-finger table, and a class cache table. Let D_p be the identifier of peer p and $D_p = P_p \cdot 2^n + S_p$. The ith entry in the c-finger table that has m entries at peer p contains the identifier of peer q in the cluster that succeeds $P_p \cdot 2^n$ by at $2^{i-1} \cdot 2^n$ on the inter-cluster identity

Table 2 Definition of data structures for peer p, using $(m+n)$-bit identifiers

Notation			Definition
c-finger table	c-finger[i]	identifier	$(P_p + 2^{i-1}) \bmod 2^m \cdot 2^n$
		node	$q = c$-successor$((P_p + 2^{i-1}) \bmod 2^m \cdot 2^n)$, where $1 \le i \le m$
	c-successor		a peer in the next cluster
	c-predecessor		a peer in the previous cluster
l-finger table	l-finger[i]	identifier	$(S_p + 2^{i-1}) \bmod 2^n$ peer $q = P_p \cdot 2^n + S_q$
		node	and $S_q = l$-successor$((S_p + 2^{i-1}) \bmod 2^n)$, $1 \le i \le n$
	l-successor		the next peer in the same cluster
	l-predecessor		the previous peer in the same cluster
class cache table	class identifier		the class identifier of data searched recently
	node		A peer in the cluster data information stores

circle, i.e., $q = c$-successor$((P_p + 2^{i-1}) \bmod 2^m \cdot 2^n)$, where $1 \le i \le m$. We call peer q the ith c-finger of peer p, and denote it by $p.c$-finger[i]. The ith entry in the l-finger table that has n entries at peer p contains the identifier of peer q whose suffix identifier S_q succeeds S_p by at 2^{i-1} on the intracluster identity circle, i.e., $S_q = l$-successor$((S_p + 2^{i-1}) \bmod 2^n)$ and $q = P_p \cdot 2^n + S_q$, where $1 \le i \le n$. Now we call peer q the ith l-finger of peer p, and denote it by $p.l$-finger[i]. And *a class cache table* entry includes both the class identifier of data searched recently and the IP address (and port number) of peer where data information stores (see Table 2).

Denote by Min(x) the node whose id is the smallest in the cluster where peer x is. Let $\{x\}$ be the node set of all the nodes in the cluster where peer x is. Let {Uinta} be the node set of all the nodes in the Uinta. Let Next$(k) = (k + 2^n) \bmod 2^{m+n}$. Then we have

$$l\text{-successor}(k) = \begin{cases} \text{Min}(P_k) & P_k \neq P_{\text{successor}(k)}, \{P_k\} \neq \emptyset, \\ \text{successor}(k) & P_{\text{successor}(k)}, \{P_k\} \neq \emptyset, \\ \text{null} & \{P_k\} = \emptyset, \end{cases} \tag{1}$$

$$c\text{-successor}(k) = \begin{cases} l\text{-successor}(\text{Next}(k)) & \{\text{Next}(k)\} \neq \emptyset, \\ c\text{-successor}(\text{Next}(k)) & \{\text{Next}(k)\} = \emptyset, \{\text{Uinta}\} \neq \emptyset, \\ \text{null} & \{\text{Uinta}\} = \emptyset, \end{cases} \tag{2}$$

$$l\text{-predecessor}(k) = \begin{cases} \text{Min}(P_k) & P_k \neq P_{\text{predecessor}(k)}, \{P_k\} \neq \emptyset, \\ \text{predecessor}(k) & P_k = P_{\text{predecessor}(k)}, \{P_k\} \neq \emptyset, \\ \text{null} & \{P_k\} = \emptyset, \end{cases} \tag{3}$$

$$c\text{-predecessor}(k) = \begin{cases} l\text{- predecessor}(\text{Next}(k)) & \{\text{Next}(k)\} \neq \emptyset, \\ c\text{-predecessor}(\text{Next}(k)) & \{\text{Next}(k)\} = \emptyset, \{\text{Uinta}\} \neq \emptyset, \\ \text{null} & \{\text{Uinta}\} = \emptyset. \end{cases} \tag{4}$$

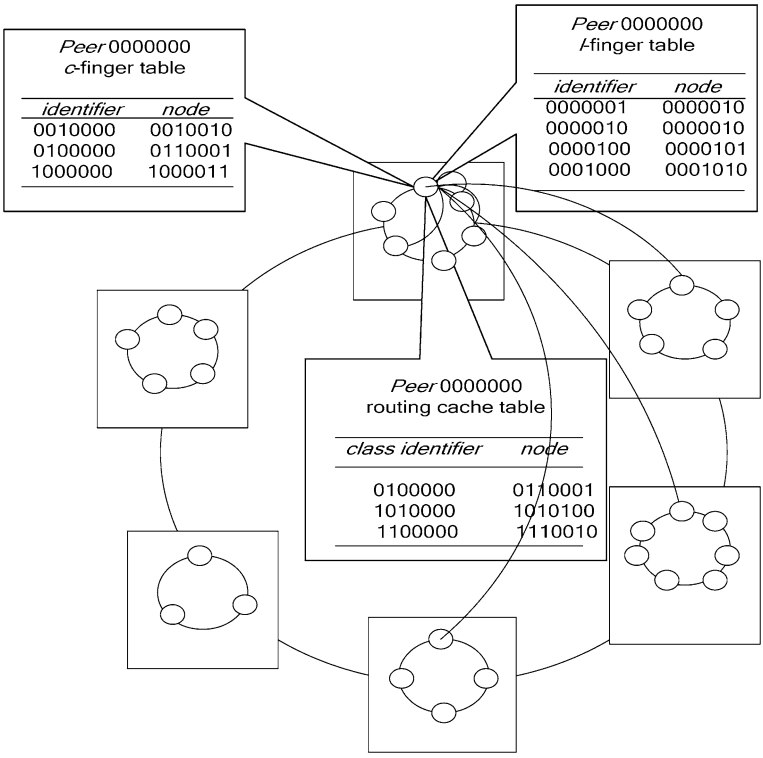


Fig. 2 An illustrative example of Uinta

Besides three tables depicted above, in Uinta, each peer should use the *landmark table* to maintain landmark nodes information, it simply records the IP addresses of all landmark nodes, which can help peer joining decide which cluster it should be located in.

Figure 2 shows an example of Uinta (with $m = 3$, $n = 4$). As shown in the figure, the search space is partitioned into 6 clusters after a series of peer joins and leaves. Peer 0 in cluster 0 maintains three tables: c-finger table, l-finger table, and class cache table. The first entry in the l-finger table of peer 0 points to peer 2 because peer 2 is the first node that succeeds within cluster 0. Similarly, the first entry in the c-finger table of peer 0 points to peer 18 because peer 18 is decided by formula (2). And the class cache table can be established after searching. From this table, we can know the entry of data class 5 is peer 84, which cannot get from the c-finger table directly.

Cache scheme Caching scheme is one of the most important aspects which differs Uinta from other P2P systems. OceanStore [24] and CFS [25] also use cache to improve system performance, which files are cached along the routing path. Because of the large storage requirement for caching files and blocks, an individual node cannot cache many files or blocks. Thus, they cannot anticipate a high cache hit rate. Such a caching scheme is not very efficient, especially in a large-scale dynamic system with a large amount of files being shared. In Uinta, it caches the information about data

classes rather than data, which we can hold a large amount of routing information with a relative small cache space and achieve a high cache hit rate. The foundation using the class cache table is that P2P system workload has temporal and spatial localities. The user tends to search data that he is interested in, which always has the similar semantic and belong to the same class. For example, a user who retrieves a song is likely to retrieve other songs in subsequential requests. Thus, the user can know which cluster it stores directly from the class cache table in the next request that searches another song, and then a significant fraction of searching will be intracluster transfers, which bypass intercluster transfers and can generate a more efficient routing algorithm.

3.2 SW-Uinta(small-world): a small-world P2P overlay network

In this section, we firstly provide the background of a well-known small-world model and its important properties. Then we present how to construct and maintain a small world P2P network SW-Uinta(small-world) based on SW-Uinta. In this paper, SW-Uinta is defined to the system which employs the nondeterministic caching scheme based on Uinta. SW-Uinta(small-world) is the system which employs the SW cache replacement scheme based on SW-Uinta.

Small world model The notion of small world phenomenon originates from social science research by Milgram [10]. He sought to determine whether most pairs of people in society were linked by short chains of acquaintances. And through some experiments, he concluded his research by showing that most pairs of people are joined by a median number of six steps, a so-called "six degrees of separation" principle.

In the following, a theoretical model for small-world networks by Watts and Strogatz [11] pictured a small world as a loosely connected set of highly connected subgraphs. The edges of the network are divided into "local" and "long-range" contacts, which are constructed roughly as follows. One starts with a set V of n points spaced uniformly on a circle, and joins each point by an edge to each of its k nearest neighbors, for a small constant k. These are the "local contacts" in the network. One then introduces a small number of edges in which the endpoints are chosen uniformly at random from V—the "long-range contacts." However, according to the model of Watts and Strogatz, there is no decentralized algorithm capable of constructing paths of small expected length [13].

Therefore, Kleinberg defined an infinite family of network models that naturally generalized the model in [11] and then proved that there was exactly one model within this family for which a decentralized algorithm existed to find short paths with high probability. In this model, the probability of a random shortcut being a distance x away from the source is proportional to $1/x$ in one dimension.

Now, it has been observed that the small world phenomenon is pervasive in a wide range of settings such as social communities, biological environments, and data/communication networks. For example, recent studies [26] have shown that P2P networks such as Freenet may exhibit small world properties. Generally, small world networks can be characterized by average path length between two nodes in the network and cluster coefficient defined as the probability that two neighbors of a node are

neighbors themselves. A network is said to be small world if it has a small average path length (i.e., similar to the average path length in random networks) and large cluster coefficient (i.e., much greater than that of random networks). Studies on a spectrum of networks with small world characteristics show that searches can be efficiently conducted when the network exhibits the following properties: (1) each node in the network knows its local neighbors, called short range contacts; and (2) each node knows a small number of chosen distant nodes, called long range contacts, with probability proportional to $1/x$ where x is the distance.

Construction Though the lookup performance can be improved in Uinta, a deterministic caching strategy is employed in it, which only achieves $O(\log N)$ search time with $O(\log N)$ cache size and maintenance cost for updating the routing table is $O(\log^2 N)$. The P2P network is the high dynamic system so that too much maintenance cost will reduce the global performance of the system. Now, we employ the small world model to construct an overlay network SW-Uinta(small-world) to get $O((\log^2 N)/k)$ search time with $O(k)$ cache size and maintenance cost for the routing table can be reduced.

In Uinta, each peer maintains two finger tables: c-finger table and l-finger table, and a class cache table. The deterministic caching strategy is employed for c-finger table and l-finger table and LRU replacement cache scheme is used for the class cache table. In SW-Uinta, each peer also maintains three cache tables. However, a nondeterministic caching strategy is proposed for two finger tables and in SW-Uinta(small-world), a cache replacement scheme related to the small-world model (SW cache replacement scheme) is used for all three cache tables based on SW-Uinta.

In the c-finger table of SW-Uinta, each peer maintains two short links: c-successor which points to the peer in the next cluster and c-predecessor which points to the peer in the previous cluster. And each peer maintains m long links c-finger[i] ($1 \le i \le m$). In the l-finger table of SW-Uinta, each peer maintains two short links: l-successor which points to the next peer in the same cluster and l-predecessor which points to the previous peer in the same cluster. And each peer maintains m long links l-finger[i] ($1 \le i \le m$). These short links are all decided by formula (1–4) and long links are decided as follows.

If the cache size for peer P is m and its cache table is full, assuming that the cache table $CT = \{d_1, d_2, \ldots, d_m\}$ and d_i is the distance between the ith cache object and P, the cache object with distance d_i is replaced by the new object with the probability $\frac{1}{D} \cdot \frac{1}{d_{m+1}}$ where $D = \sum_{i=1}^{m+1} 1/d_i$ when a new object with distance d_{m+1} is received. We call this scheme SW cache replacement scheme.

Suppose that peer S gets the answer which it requests for from peer T and there are no pointers to peer T in the cache table of peer S.

(1) If peer S and peer T are in the same cluster and l-finger[i] is not full, peer S caches peer T in the l-finger table;
(2) If peer S and peer T are in the same cluster and l-finger[i] is full, SW cache replacement scheme is employed for l-finger[i];
(3) If peer S and peer T are in the different clusters and c-finger[i] is not full, peer S caches peer T in the c-finger table;

(4) If peer S and peer T are in the different clusters and c-finger[i] is full, SW cache replacement scheme is employed for c-finger[i].

Because users always retrieve data of a kind during a period of time, which they are interested in, we store the data information based on data semantics in Uinta, which makes data of a kind placed in the same cluster in Uinta. A high hit rate for the class cache table can be expected, thus a reduced average number of routing hops and lower routing network latency can be achieved. In Uinta, we used LRU as the cache replacement scheme, which could not improve the whole performance because the network characteristic is not considered in LRU. Therefore, in SW-Uinta, we employ the SW cache replacement scheme based on the small world model. We call this system SW-Uinta(small-world). When peer S gets data D which it requests from peer T, the class identifier of data D and the IP address (and port number) of some peer in the cluster where T is will be stored in the class cache table if it is not full. Otherwise, if it is full, SW cache replacement scheme is employed for the class cache table.

Cache tables are generally kept fresh by the traffic of requests traveling through peers. To handle pathological cases in which there are no lookups for a particular ID range, each peer refreshes any cache table to which it has not performed data lookup in the past hour. Refreshing means picking an entry in the cache table and performing a peer search for that ID. On the one hand, this scheme avoids the bottleneck of network traffic because all of the peers will not update at the same time. On the other hand, the peer can be response for the failure of some peers quickly. At last, maintenance cost for the routing table always goes with the lookup operation, which needs few other messages. In the worst case, that is, most of the cache table entries have not been used to lookup the data. There is no an obvious advantage for the SW cache replacement scheme compared with the LRU scheme. However, this situation is highly unlikely in a real world application.

4 Protocols for the SW-Uinta(small-world) overlay network

In this section, several protocols are provided for the SW-Uinta(small-world) network, such as peer join/leave protocol and routing protocol.

4.1 Peer join/leave protocols

Peer joins When a new peer p wants to join the system, it must send a join message to a nearby peer q that is already a member of the system. This process can be done in different methods. We just omit it and simply assume it can be done quickly. (This is the same assumption as other DHT algorithms.) After that, peer p can get the information of landmark nodes from this nearby peer q and fulfills its own landmark table. It then decides the distance between the landmark nodes and itself and employs the distributed binning scheme to determine the suitable cluster P_p it should join. In the following, the identifier D_p of peer p can be gotten, *i.e.*, $D_p = P_p \cdot 2^n + S_p$ (S_p is the hash value of IP address of peer p). Consequently, peer p connects the peer p' in the cluster P_p through the c-finger table of peer q and then is located in the cluster

based on the suffix S_p. Assume that peer s is the l-successor of peer p and peer n is the original l-predecessor of peer s. Then peer p acquires peer s as its l-successor and acquires peer n as its l-predecessor. Peer n acquires peer p as its l-successor and peer s acquires peer p as its l-predecessor. The c-predecessor node and c-successor node of peer p can be decided by formula (2) and formula (4). Other data structures needed by peer p are copied from peer s.

If peer p finds that c-finger[i].identifier is equal to P_p but the identifier prefix of c-finger[i].node denoted as peer x is not equal to P_p rather than X_p in the cache table of peer q, it shows peer p will form a new cluster whose identifier is P_p. Short links of peer p are also decided by formula (1–4). Other data structures needed by peer p are copied from peer x. Finally, keys between $P_p \cdot 2^n$ and $X_p \cdot 2^n$ are moved form cluster $X_p \cdot 2^n$ to cluster $P_p \cdot 2^n$. Peer p joins the system successfully.

Peer leaves or fails When a peer leaves the network, it checks whether it is the last peer in the cluster. If there are other peers in the cluster, this peer simply informs its leaving to its l-predecessor and l-successor and the key subspace in it are moved to its l-successor. Otherwise, except for informing its leaving to its c-predecessor and c-successor, the key subspace of this cluster needs to be merged with one of its neighboring clusters. The peer of its neighboring cluster which is closest in the key space to the leaving peer takes over all of its keys. A failed peer is detected during routine operations such as search queries. If a peer detects a peer to fail which is in one of its cache tables, it evicts this entry in the cache table.

4.2 SW-Uinta(small-world) routing protocol

SW-Uinta(small-world) routing algorithm and an example of routing are given in this section.

(1) When peer p wants to obtain the file associated with some key k and some class c, it gets the class identifier P_k of file hashed by SHA-1 with c.
(2) check whether exists an entry $(P_k \cdot 2^n, q)$ for the class identifier P_k in the class cache table; if does, jump to peer q directly, then to (6); otherwise, to (3).
(3) check whether P_k falls between the P_p of p and the P_q of its c-successor q; if does, jump to q, then to (6); otherwise, to (4).
(4) $x = p$;
    repeat
        Searches peer x's c-finger table for peer q whose prefix of identifier P_q most immediately precedes P_k, $x = q$;
        until P_k falls between the P_x of x and the P_q of its c-successor q.
(5) jump to peer q.
(6) find a peer d through the l-finger table of peer q so as to make the suffix of key identifier S_k hashed by SHA-1 with k fall between the S_x of x and the S_d of its l-successor d.
(7) return the identifier of peer d and (key, value) pair searched to peer p; join the information of peer d to two finger tables of peer p and join $(P_k \cdot 2^n, d)$ to the class cache table of peer p using the SW cache replacement scheme description above.

For example, we want to find a film named *Shrek*. Firstly, we get the class identifier $P = \text{hash(film)}$ and find the peer cluster that P exists through the cache table and c-finger table. Then we get the file identifier $S = \text{hash(Shrek)}$. Lastly, we can find the information of this film through S and l-finger table in this peer cluster.

5 Theoretical analysis

In this section, we analyze Uinta routing latency and give two properties for the SW-Uinta(small-world) network.

Assume that there are N peers in both Chord and Uinta and let M be the number of clusters in Uinta. Denote by $L_{\text{Chord-hop}}$ the average network latency for each hop (hop latency) in Chord, thus the average routing latency in Chord is:

$$L_{\text{Chord}} = \frac{1}{2} \cdot \log_2 N \cdot L_{\text{Chord-hop}}. \tag{5}$$

While in Uinta, denote by $L_{\text{Uinta-inter}}$ the average network latency for each hop between the clusters in Uinta and denote by $L_{\text{Uinta-intra}}$ the average network latency for each hop within the cluster in Uinta and there are N_i peers in cluster i, thus the average routing latency in Uinta is:

$$L_{\text{Uinta}} = \frac{1}{2} \cdot \log_2 M \cdot L_{\text{Uinta-inter}} + \frac{1}{M}\frac{1}{2} \cdot \log_2 \prod_{i=1}^{M} N_i \cdot L_{\text{Uinta-intra}}. \tag{6}$$

In our simulations, we found the intercluster hop latency in Uinta is nearly the same or slightly larger than the hop latency in Chord and the intracluster hop latency is much smaller than the hop latency in Chord and $N \gg M$. That is,

$$\begin{aligned} L_{\text{Chord-hop}} &\approx L_{\text{Uinta-inter}}, \\ L_{\text{Chord-hop}} &> L_{\text{Uinta-intra}} \end{aligned} \tag{7}$$

and

$$\because \quad \sqrt[M]{N_1 N_2 \cdots N_m} \le \frac{1}{M}(N_1 + N_2 + \cdots + N_m). \tag{8}$$

Then we have

$$\begin{aligned} L_{\text{Uinta}} &\le \frac{1}{2} \cdot \log_2 M \cdot L_{\text{Uinta-inter}} + \frac{1}{2M} \cdot \log_2 \left(\frac{N}{M}\right)^M \cdot L_{\text{Uinta-intra}} \\ &= \frac{1}{2} \cdot \log_2 M \cdot L_{\text{Uinta-inter}} + \frac{1}{2} \cdot \log_2 \frac{N}{M} \cdot L_{\text{Uinta-intra}} \\ &< \frac{1}{2} \cdot \left(\log_2 M + \log_2 \frac{N}{M}\right) \cdot L_{\text{Uinta-inter}} \\ &\approx \frac{1}{2} \cdot \log_2 N \cdot L_{\text{Chord-hop}} = L_{\text{Chord}}. \end{aligned} \tag{9}$$

So we can expect a routing deduction by using the routing algorithm in Uinta. Suppose in a P2P system with 2^{20} nodes and the average latency per hop in Chord algorithm is 100 ms and the average latency between the clusters in Uinta is 108 ms. Then the average routing latency in Chord algorithm is equals 1,000 ms. Assuming that all of peers are formed 2^{10} clusters in a Uinta system, the average latency within the cluster is only half of the latency between two clusters which is 54 ms each hop, thus the average routing network latency in Uinta is less than which is approximately to 810 ms. The system average routing latency reduces by 19%. If we consider the cache scheme using in Uinta and denote by P the hit ratio, we can get

$$\begin{aligned} L_{\text{Uinta-cache}} &\le P\left(1 \cdot L_{\text{Uinta-inter}} + \frac{1}{2}\log_2 \frac{N}{M} \cdot L_{\text{Uinta-intra}}\right) \\ &\quad + (1-P)\left(\frac{1}{2} \cdot \log_2 M \cdot L_{\text{Uinta-inter}} + \frac{1}{2} \cdot \log_2 \frac{N}{M} \cdot L_{\text{Uinta-intra}}\right) \\ &= \left[P + \frac{1}{2}(1-P)\log_2 M\right] \cdot L_{\text{Uinta-inter}} + \frac{1}{2} \cdot \log_2 \frac{N}{M} \cdot L_{\text{Uinta-intra}} \\ &\le \frac{1}{2}\log_2 M \cdot L_{\text{Uinta-inter}} + \frac{1}{2} \cdot \log_2 \frac{N}{M} \cdot L_{\text{Uinta-intra}} \\ &< L_{\text{Uinta}}. \end{aligned} \tag{10}$$

So we can reduce the more routing latency using the cache scheme. Assuming P is 40%, thus $L_{\text{Uinta-cache}}$ in Uinta is less than which is approximately to 637 ms, which reduce the latency by 36%.

From above description, we can know that the lookup performance can significantly be improved in Uinta where the physical topology of network and data semantics are both considered when constructing it.

Now two properties for the SW-Uinta(small-world) and their proofs are given in the following.

Property 1 Employing the SW cache replacement scheme, the probability of having the key K' in the cache of peer K tends to be proportional to $1/|K - K'|$.

Proof We define a Markov chain as follows. If the distance between the cache object and the peer's own key K is x, we say that the Markov chain is in state x. Then the state place for peer K is $ST = \{d_1, d_2, \ldots, d_m\}$ and d_m is the distance between the cache object and K.

According to the SW cache replacement scheme, we can get the transition probability matrix is

$$P = \begin{bmatrix} \frac{\frac{1}{d_1}}{D} & \frac{\frac{1}{d_2}}{D} & \cdots & \frac{\frac{1}{d_m}}{D} \\ \frac{\frac{1}{d_1}}{D} & \frac{\frac{1}{d_2}}{D} & \cdots & \frac{\frac{1}{d_m}}{D} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\frac{1}{d_1}}{D} & \frac{\frac{1}{d_2}}{D} & \cdots & \frac{\frac{1}{d_m}}{D} \end{bmatrix}$$

where $D = \sum_{i=1}^{m} \frac{1}{d_i} \cdot p_{ij}$ refers to the probability that d_i is replaced by d_j.

Because the Markov chain is irreducible and aperiodic with the finite number, a stationary probability distribution for the process exists. Let's call π_x the stationary probability of residing in state x. Then

$$\begin{cases} \pi_i = \frac{\frac{1}{d_i}}{D} \sum_{i=1}^{m} \pi_i & (i = 1, 2, \ldots, m). \\ \sum_{i=1}^{m} \pi_i = 1 \end{cases}$$

Hence, $\pi_i = \frac{1}{D} \cdot \frac{1}{d_i}$ satisfies the equation for all i and the stationary solution is unique.

Therefore, employing our cache replacement scheme, the probability of having the key K' in the cache of peer K tends to be proportional to $1/|K - K'|$. According to the property of the Markov chain, no matter what the initial keys we assigned to the nodes, the system forms toward the desire solution as soon as enough random queries are made. □

Therefore, from Property 1, we can know SW-Uinta(small-world) satisfies the small-world model.

Property 2 In SW-Uinta(small-world), the expected number of hops required to lookup an object is $O((\log^2 N)/k)$.

Proof If there are M peers in the system which has a ring topology, for $j > 0$, we say it is in phase j when the distance from the current peer S to the destination peer T is greater 2^j and at most 2^{j+1}. We say it is in phase 0 when the distance is at most 2. Hence, the maximum number of times the original distance could possible be halved before it is at most 2 is $1 + \log M$.

Assume that in SW-Uinta(small-world), there are 2^m clusters and the average number of peers in each cluster is 2^n. Then the total number of peers in the system $N = 2^{m+n}$. Identifiers are represented as a circle of numbers from 0 to $N - 1$. The current peer S chooses a contact that is as close to the target peer T as possible. Therefore, the maximum number of lookup times among the clusters is $1 + \log 2^m = 1 + m$ and the maximum number of lookup times in each cluster is $1 + \log 2^n = 1 + n$.

According to Property 1, the probability distribution function is $p(x) = \frac{1}{D} \cdot \frac{1}{x}$ when x lies in the range $[1, 2^m]$ where $D = \sum_{x=1}^{2^m - 1} \frac{1}{x} \leq 1 + \ln(2^m - 1) \leq \ln(2^{m+2})$. The probability p_{half} of drawing a value from $[z/2, z]$ for any $z \in [2, 2^m]$ among the clusters is given by

$$\int_{z/2}^{z} p(x)\, dx = \int_{z/2}^{z} \frac{1}{D} \frac{1}{x}\, dx \geq \int_{z/2}^{z} \frac{1}{\ln N} \frac{1}{x}\, dx = \frac{1}{\log_2 2^{m+2}},$$

which is independent of z. The number of links to consider before the current distance diminishes by at least half follows a geometric distribution with $\mu = \frac{1}{p_{\text{half}}} = \log_2 2^{m+2}$. With k links per peer, the expected number of clusters to consider before the current cluster distance is at least halved is at most $2 \log_2 2^{m+2}/k = 2(m + 2)/k$.

Thus, the average path length among the clusters is at most $(1+m)2(m+2)/k$. At the same time, the average path length in the cluster is at most $(1+n)2(n+2)/k$. So the average path length of the full route is at most

$$(1+m)2(m+2)/k + (1+n)2(n+2)/k \cdot 2[(m+2)^2 + (n+2)^2]/k \cdot 2(m+n+4)^2/k = 2(\log N + 4)^2/k.$$

Therefore, in SW-Uinta(small-world), the expected number of hops required to lookup an object is $O((\log^2 N)/k)$. □

From the description of two properties, we can know the SW-Uinta(small-world) overlay network satisfies its constructing aim.

6 Performance evaluations

In this section, we describe the simulation methodology and several performance metrics firstly. Then we present simulation results showing the routing performance improvement possible with the use of SW-Uinta(LRU), SW-Uinta(small-world), Uinta-cache $\log N$ compared with Chord.

6.1 Simulation methodology and performance metrics

In our simulation, we used the GT-ITM [27] transit stub topology generator to generate the underlying networks, which the number of system peers N is varied from 1,000 to 10,000. As far as the logical overlay is concerned, we build SW-Uinta(LRU), SW-Uinta(small-world) and Uinta based Chord simulator. Each peer on the overlay is uniquely mapped to one node in the IP layer. The selection of landmark nodes has great impact on the performance of system. It affects the network latency measurement accuracy and has the direct influence on node clustering decision. Increasing the number of landmarks improves the accuracy of destination selection, with decreasing returns as the set size increases. Thus, a moderate number of landmarks greatly improve performance. Moreover, all of the landmarks must satisfy an even distribution. So we choose 4 landmark nodes, each for an intratransit field at random and there is 3-level latency from landmark nodes to peers. $100 \cdot N$ pseudo file-ids that are classified into 100 categories are generated and distributed across all the peers in the simulated network. For each experiment, 100,000 randomly generated routing requests (including file-id and its class) are executed. To obtain a fair comparison, we keep the size of the cache table in Uinta, SW-Uinta(LRU) and SW-Uinta(small-world) to be also $O(\log N)$. We choose Chord as the platform because the ring geometry allows the greatest flexibility.

We consider four metrics to verify the effectiveness of our systems: (1) Routing hop: the average number of logical hops traversed by search messages to the destination; (2) Routing latency: the average time for search messages from the source to the destination; (3) Latency stretch: the ratio of the average latency on the overlay network to the average latency on the IP network; (4) Maintenance cost: the average

number of messages incurred for each peer joining/leaving and for RT maintenance cost which is the average number of messages needed for maintaining the routing table to be up-to-date.

6.2 Routing cost reduction

The goal of simulation in this section is to show whether SW-Uinta(LRU), SW-Uinta(small-world) and Uinta can reduce the routing cost in P2P system. Figure 2 shows the results of routing hops and routing latency evaluation. In this simulation, we compare the routing performance of Uinta-cache $\log N$, SW-Uinta(LRU) and SW-Uinta(small-world) with Chord under different size networks. SW-Uinta(LRU) refers to the system which employs the LRU cache replacement scheme based on SW-Uinta. SW-Uinta(small-world) is the system which employs the SW cache replacement scheme based on SW-Uinta.

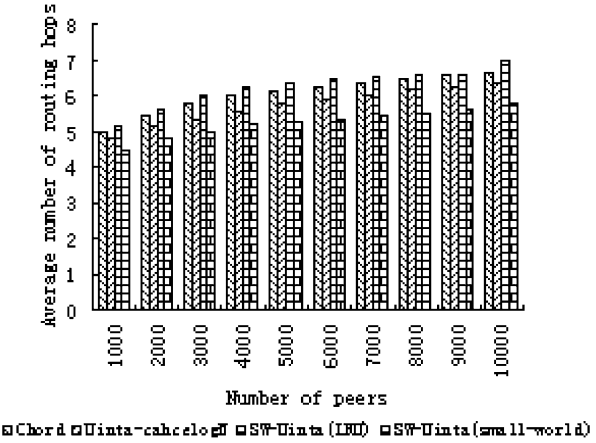
Figure 3(a) shows the routing performance comparison result measured with the average number of routing hops. Chord, Uinta-cache $\log N$, SW-Uinta(LRU) and SW-Uinta(small-world) all have good scalability: with the network size increasing from 1,000 nodes to 10,000 nodes, the average number of routing hops only increases around 33%, 32%, 36%, and 29%, respectively, and the average number of routing hops is 6.07, 5.74, 6.24, and 5.24, respectively.

As a proximate metrics, the average number of routing hops cannot represent the real routing cost. The actual routing latency is highly depended on the average latency for each hop. Figure 3(b) shows the measured results of the average routing latency for Chord, Uinta-cache $\log N$, SW-Uinta(LRU) and SW-Uinta(small-world) algorithms. Although Uinta, SW-Uinta(LRU) and SW-Uinta(small-world) have the nearly equal average number of routing hops with that of Chord, they have smaller average routing latency which can represents the real routing cost. For Uinta-cache $\log N$, SW-Uinta(LRU) and SW-Uinta(small-world), the average routing latency gets 24.3%, 18.0%, and 40.4% reduction, respectively, compared with Chord.

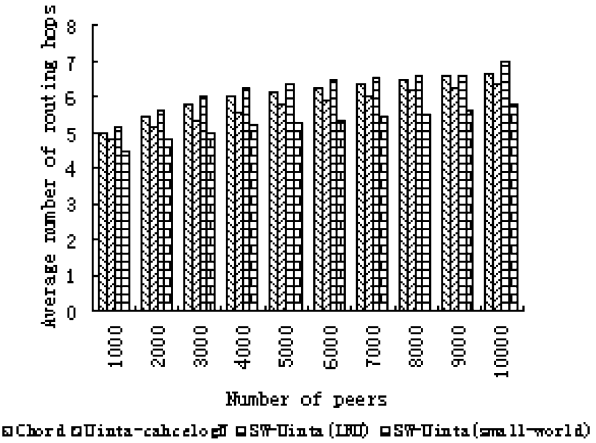
Obviously, the performance of routing hops and scalability for SW-Uinta(LRU) are worse than those of other algorithms, which shows the LRU cache replacement is not suitable for the P2P system. At the same time, because both the network characteristic and the network topology are considered in constructing the SW-Uinta(small-world) system so that the routing performance of SW-Uinta(small-world) is better than that of three other algorithms.

6.3 Stretch reduction

Latency stretch is defined as the ratio of the average latency on the overlay network to the average latency on the IP network, which can be used to characterize the match degree of the overlay to the physical topology. Table 3 summarizes the stretch statistics in the case of a 10,000-peer network. According to it, we can know that stretch can be reduced using Uinta, SW-Uinta(LRU) and SW-Uinta(small-world). This shows that using topology-aware and semantic-aware overlay construction, we can achieve significant improvement in the lookup performance. And SW-Uinta(small-world) can get better performance than Uinta.



(a) Average number of routing hops



(b) Average routing latency

Fig. 3 Routing performance comparison for Uinta, SW-Uinta and Chord

Table 3 Latency stretch result for Chord, Uinta and SW-Uinta

Algorithm	Average routing latency	Latency stretch
Chord	531.51	4.40
Uinta-cache log N	366.68	3.04
SW-Uinta(LRU)	397.28	3.29
SW-Uinta(small-world)	289.31	2.40

6.4 Maintenance cost

In this experiment, we show that SW-Uinta(small-world) not only keeps the strengths of Uinta, but also is able to reduce maintenance cost. Maintenance cost here includes two parts: the number of messages incurred for each peer joining/leaving and for RT maintenance cost. So maintenance cost is defined as the average number of messages

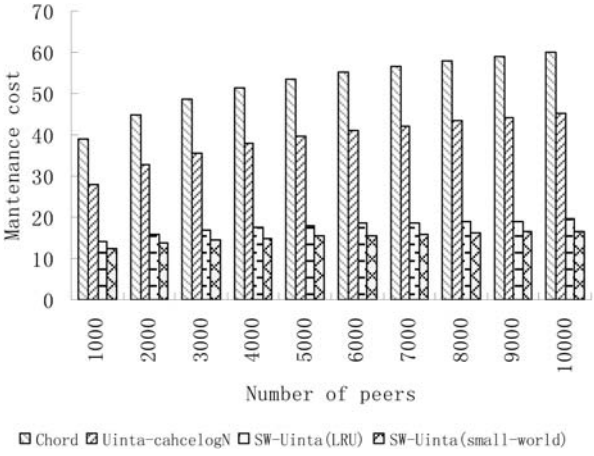


Fig. 4 Maintenance cost comparison for Uinta, SW-Uinta and Chord

for all maintaining operations such as peer joining, peer leaving, and keeping the routing table to be up-to-date.

We assume that peer joins/leaves according to the Poisson process at rate $R = 0.1$ (min^{-1}). RT maintenance period is an hour. Figure 4 depicts maintenance cost comparison for the Uinta-cache $\log N$, SW-Uinta(LRU), SW-Uinta(small-world) and Chord schemes under the different network size. Maintenance cost is around 16, 7, 6, and 21, respectively, for the Uinta-cache $\log N$, SW-Uinta(LRU), SW-Uinta(small-world), and Chord schemes. From this experiment, we know maintenance cost for SW-Uinta is lower than two other algorithms because there is no additional cost for the route table maintenance, which is RT maintenance cost, in SW-Uinta. Each search operation can update the routing table, which needs no other operations to maintain the routing table to be up-to-date.

7 Conclusions and future works

In this paper, we propose a new overlay infrastructure SW-Uinta(small-world) based on Uinta. SW-Uinta(small-world) not only holds the strength of Uinta, which takes both the user's interest and the physical topology into consideration, but also considers the network character so that the SW cache replacement scheme is proposed to further improve the performance of object lookup. In addition, the P2P system is a dynamic environment, maintenance cost for peer joining, peer leaving, and routing state maintaining is a very high. Therefore, we propose a nondeterministic caching scheme to reduce maintenance cost when peers self-organization occurs. Simulations also show SW-Uinta(small-world) can improve the lookup performance as well as it can reduce maintenance cost under the same size of routing table. However, this infrastructure is only suitable for key-based retrieval now and content-based retrieval is our next step work.

References

1. Gnutella Website. http://www.gnutellaforums.com/
2. Leibowitz N, Ripeanu M, Wierzbicki A (2003) Deconstructing the Kazaa network. In: Proceedings of 3rd IEEE workshop on Internet applications, Santa Clara, CA, pp 112–120
3. Clarke I, Sandberg O, Wiley B et al (2000) Freenet: A distributed anonymous information storage and retrieval system. In: Proceedings of workshop on design issues in anonymity and unobservability, ICSI, pp 311–320
4. Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H (2001) Chord: a scalable peer-to-peer lookup service for Internet applications. In: Proceedings of the ACM SIGCOMM. ACM Press, New York, pp 149–160
5. Ratnasamy S, Francis P, Handley M, Karp R, Shenker S (2001) A scalable content-addressable network. In: Proceedings of ACM SIGCOMM. ACM Press, New York, pp 161–172
6. Rowstron A, Druschel P (2001) Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Proceedings of the 18th IFIP/ACM international conference on distributed systems platforms. Springer, Berlin, pp 329–350
7. Zhao BY, Huang L, Stribling J, Rhea J, Joseph SC, Kubiatowicz AD (2004) Tapestry: a resilient global-scale overlay for service deployment. IEEE J Sel Areas Commun 22:41–53
8. Ratnasamy S, Handley M, Karp R et al (2002) Topologically-aware overlay construction and server selection. In: Proceedings of IEEE INFOCOM'02. IEEE Press, New York, pp 1190–1199
9. Mahanti A (1999) Web proxy workload characterization and modeling. Master's thesis, Department of Computer Science, University of Saskatchewan, September 1999
10. Milgram S (1967) The small world problem. Psychol Today 2:60–67
11. Watts D, Strogatz S (1998) Collective dynamics of small-world networks. Nature 393:440–442
12. Kleinberg J (2002) Small-world phenomena and the dynamics of information. In: Proceedings of advances in neural information processing systems. MIT Press, Cambridge, pp 14–25
13. Kleinberg J (2000) The small-world phenomenon: an algorithmic perspective. Technical Report, Cornell Computer Science
14. Iamnitchi A, Ripeanu M, Foster I (2004) Small-world file-sharing communities. In: Proceedings of IEEE INFOCOM. IEEE Press, New York, pp 175–186
15. Maymounkov P, Mazieres D (2002) Kademlia A peer-to-peer information system based on the xor metric. In: Proceedings of the 1st international workshop on peer-to-peer systems (IPTPS'02), Cambridge, MA, pp 53–65
16. Malkhi D, Naor M, Ratajczak D (2002) Viceroy: a scalable and dynamic lookup network. In: Proceedings of the 21st ACM symposium on principles of distributed computing PODC'02, Monterey, CA, pp 183–192
17. Manku GS, Bawa M, Raghavan P (2003) Symphony: distributed hashing in a small world. In: Proceedings of the fourth USENIX symposium on Internet technologies and systems (USITS), Seattle, WA, pp 127–140
18. Roussopoulos M, Baker M (2003) CUP: controlled update propagation in peer-to-peer networks. In: Proceedings of the 2003 USENIX annual technical conference
19. Garces-Erice L, Ross KW, Biersack EW, Felber PA, Urvoy-Keller G (2003) Topology-centric look-up service. In: Proceedings of COST264/ACM fifth international workshop on networked group communications (NGC), Munich, Germany. Springer, Berlin
20. Harvey NJA, Theimer M, Jones MB, Sarouiu S, Wolman A (2003) Skipnet: a peer-to-peer overlay network. In: Proceedings of the fourth USENIX symposium on Internet technologies and systems
21. Freedman MJ, Freudenthal E, Mazières D (2004) Democratizing content publication with coral. In: Proceedings of USENIX/ACM symposium on networked systems design and implementation (NSDI'04), San Francisco, CA
22. Gummadi K, Gummadi R, Gribble S, Ratnasamy S, Shenker S, Stoica I (2003) The impact of DHT routing geometry on resilience and proximity. In: Proceedings of ACM SIGCOMM
23. Karger DR, Lehman E, Leighton F et al (1997) Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In: Proceedings 29th annual ACM symposium theory of computing, El Paso, TX, pp 654–663
24. Kubiatowicz J, Bindel D, Eaton P et al (2000) OceanStore: an architecture for global-scale persistent storage. In: Proceedings of the 9th international conference on architectural support for programming languages and operating systems (ASPLOS'00), Cambridge, MA, pp 190–201

25. Dabek F, Frans Kaashoek M, Karger D et al (2001) Wide-area cooperative storage with CFS. In: Proceedings of the 18th ACM symposium on operating systems principles (SOSP'01), Banff, Alberta, Canada, pp 202–215
26. Zhang H, Goel A, Govindan R (2002) Using the small-world model to improve Freenet performance. In: Proceedings of IEEE INFOCOM 2002. IEEE Press, New York, pp 1228–1237
27. Zegura EW, Calvert K, Bhattacharjee S (1996) How to model an Internet work. In: Proceedings of the 15th annual joint conference of the IEEE computer and communications societies (INFOCOM'96). IEEE Communications Society, San Francisco, pp 594–602

Jie Xu is a lecturer of faculty of mathematics and computer science, Hubei University, Wuhan, China. She received Ph.D. degree at HUST in 2007. Her research interests include peer to peer systems, operating system, and fault tolerance computing.

Hai Jin is a professor and dean of school of computer science and technology, Huazhong University of Science and Technology (HUST), Wuhan, China. He received Master and Ph.D. degrees at HUST in 1991 and 1994, respectively. He was a Postdoc Fellow at Department of Electrical and Electronic Engineering, University of Hong Kong, and a visiting scholar at Department of Electrical Engineering-System, University of South California, Los Angeles, USA, from 1998 to 2000. His research interests include cluster computing, grid computing, peer to peer systems, multimedia systems, network storage, and network security. He is the editor of several international journals, such as International Journal of Computers and Applications, International Journal of Grid and Utility Computing, Journal of Computer Science and Technology. He is now leading the largest grid computing project in China, called ChinaGrid, funded by the Ministry of Education, China.