A Virtual Ring Model for Constructing and Analyzing Structured Small-World Networks Postprint

Authors: Zhuge Hai, Sun Xiaoping

Date: 2016-11-02T00:00:00+00:00

Abstract

Constructing structured peer-to-peer networks that support efficient information query requires consideration of both the characteristics of the underlying network topology and distance metrics in the data space. Small-world models can help us establish a universal model for constructing structured peer-to-peer networks, enabling the design of efficient routing methods based on distance metrics in the data space. This paper introduces a virtual ring model that constructs long links tailored to different distance metric methods and underlying network topologies, forming a small-world topology that supports efficient deterministic greedy routing. By leveraging the four properties provided by the virtual ring model, one can analyze whether the underlying topology of a structured peer-to-peer network can support effective greedy routing through the addition of long links by mapping the underlying network topology onto a virtual ring network. The virtual ring model provides corresponding topology analysis methods and routing table long link construction methods for two major categories of network underlying topologies. This paper applies the virtual ring model to add long links and construct structured small-world networks supporting deterministic greedy routing in ring networks based on ring distance, d-dimensional torus networks based on Manhattan distance, and ring networks based on tree classification distance. The paper also discusses other networks such as De Bruijn networks and routing hops in dynamic scenarios. Theoretical analysis and experimental validation demonstrate the effectiveness of the network topology and routing methods constructed based on this model. As a relatively universal and practical routable small-world model, the virtual ring model provides a systematic foundational theoretical and methodological framework for implementing efficient distributed routing oriented toward information retrieval queries in data spaces with specific distance metrics.

Full Text

Preamble

Vol.7 No.2 Information Technology Letters Vol.7 No.2
A Virtual Ring Model for Building and Analyzing Structured Small-World Networks
Zhuge Hai, Sun Xiaoping

Abstract: Building structured peer-to-peer networks that support efficient information query requires consideration of both the underlying network topology characteristics and distance metrics in the data space. Small-world models can help us establish a general framework for constructing structured peer-to-peer networks, designing efficient routing methods based on distance metrics in the data space. This paper introduces a virtual ring model that, for different distance metrics and underlying network topologies, constructs long links to form small-world topologies supporting efficient deterministic greedy routing. Using the four properties provided by the virtual ring model, we can analyze whether the underlying topology of a structured peer-to-peer network can support effective greedy routing by mapping it to a virtual ring network. The virtual ring model provides corresponding topology analysis methods and routing table long-link construction methods for two major classes of network underlying topologies. This paper applies the virtual ring model to add long links on ring networks based on ring distance, d-dimensional torus networks based on Manhattan distance, and ring networks based on tree classification distance, constructing structured small-world networks that support deterministic greedy routing. The paper also discusses routing hops in other networks such as De Bruijn networks and under dynamic conditions. Theoretical analysis and experiments verify the effectiveness of the network topology and routing methods constructed based on this model. As a relatively general and practical routable small-world model, the virtual ring model provides a systematic theoretical framework for implementing efficient distributed routing for information retrieval queries in data spaces with specific distance metrics.

Keywords: small-world networks; peer-to-peer networks; structured; routing hops

1. Introduction

Peer-to-Peer (P2P) technology provides effective technical means for scalable data publishing, querying, and sharing in large-scale distributed environments. Among them, structured Distributed Hash Table (DHT) peer-to-peer networks

support scalable deterministic routing in large-scale distributed environments and have received widespread attention from researchers. Structured peer-to-peer networks achieve efficient distributed routing algorithms by adding long links on a specific underlying network topology [12], [20], [26], [27], [28], [36]. In structured peer-to-peer networks, nodes are assigned positions in an identifier (ID) space, and nodes closest to each other in the ID space are connected through short links to form the underlying network topology. On top of this underlying topology, each node adds several long links to support efficient routing. Typically, the number of links is logarithmic in network size, and the length of nodes' long links grows exponentially in the ID space. During routing, the current node selects the long link with the shortest distance to the target node as the next hop. This way, using only local information at each node can achieve logarithmic routing efficiency. Although researchers have proposed various structured peer-to-peer networks, most structured networks achieve routing efficiency of $O(\log n)$ when each node maintains $O(\log n)$ long links.

Researchers have proposed some models for analyzing the topology and routing characteristics of structured peer-to-peer networks [11], [17], [25], [29]. However, we still need a general theoretical analysis framework for constructing structured peer-to-peer networks based on specific underlying topologies in a given ID space with specific distance metrics, and for analyzing the corresponding routing properties. This paper introduces a general model called the virtual ring model for building and analyzing small-world networks on structured networks. The virtual ring model leverages Jon Kleinberg' s routable small-world model [14] to construct structured small-world peer-to-peer networks. The small-world phenomenon was first discovered in social networks, where letters between two strangers could be delivered through no more than six intermediaries [18]. In the late 1980s, researchers began using random graph models to analyze the diameter and clustering characteristics of small-world networks [2]. Small-world networks are generally considered to have short network diameters (logarithmic) and large clustering coefficients. Many natural and artificial networks, such as the World Wide Web, have been confirmed to possess small-world network characteristics [32]. Many unstructured peer-to-peer networks like Gnutella also have small-world network characteristics [22], but cannot support deterministic routing. In fact, most small-world network models cannot explicitly support deterministic routing—that is, quickly (in logarithmic time) discovering routing targets using local information. Kleinberg pioneeringly proposed a routable small-world model. This model adds one long link for each node on a two-dimensional grid, with the probability of building a link between two nodes inversely proportional to their distance. This model can be seen as a small-world network model supporting deterministic greedy routing strategies [14]. Kleinberg' s model provides a feasible theoretical foundation for building peer-to-peer networks. Based on Kleinberg' s model, researchers have proposed methods for constructing small-world peer-to-peer networks on various underlying topologies including rings and Delaunay graphs [Figure 2: see original paper] [4], [6], [10], [16], [19], [30].

Compared with previous work [7], [14], [15], [17], [23], [25], [29], [33], the virtual ring model can not only be used to construct routable structured small-world peer-to-peer networks in a given ID space, but also to analyze the routing efficiency of the proposed methods. The model takes the underlying topology, ID space, and distance metric in the ID space as core factors, providing methods for verifying the routability of the underlying topology, designing routing methods, and analyzing routing hops. It answers three questions: (1) Given an underlying topology network structure, ID space, and distance metric, can we support effective greedy routing by adding a certain number (logarithmic level) of long links to each node? (2) If yes, how to add them? (3) What is the routing efficiency after addition?

The remainder of this paper is organized as follows:

2. Related Work

Structured peer-to-peer networks use distributed hash tables to map both data and nodes to the same linear ID space. The publication location of data is determined by the linear order relationship between data IDs and node IDs. Node routing tables are constructed in the linear ID space to effectively support exact queries. Each node adds $O(\log n)$ links, and routing can achieve $O(\log n)$ hops. Both the underlying topology and the construction of long links in structured peer-to-peer networks are determined by the distance metric defined on the ID space. Tapestry [36] and Pastry [27] algorithms, based on the work of Plaxton et al. [24], construct routing tables based on the shared prefix of b-ary digital IDs, where the distance in ID space is the length of the shared prefix between two nodes, achieving $O(\log_b n)$ routing hops. Chord [28] constructs routing tables in an integer ID space, where the distance between two points is the numerical distance of integer IDs. The routing table size is $O(\log n)$. CAN [26] partitions a d-dimensional Cartesian coordinate space and establishes direct neighbor links for adjacent nodes in the space, supporting multi-dimensional data location queries. In CAN peer-to-peer networks, nodes uniformly partition the d-dimensional Cartesian coordinate space into multiple hypercube intervals. The distance between two points is Manhattan distance. Each node links to its $O(d)$ adjacent neighbor nodes in the space, and greedy query routing along neighbor nodes has a path length of $O(dn^{1/d})$. If d is $O(\log n)$, then $O(\log n)$ routing hops can be achieved, with each node maintaining $O(\log n)$ direct neighbors. Numerous studies have proposed theoretical analysis frameworks and models for various topology and routing characteristics of existing structured peer-to-peer networks [1], [11], [17], [25], [29], [33].

Kleinberg' s small-world model can serve as a general constructive model for building small-world peer-to-peer network topologies [14]. Previous small-world models could not directly implement deterministic greedy routing. Kleinberg' s model adds long links on a two-dimensional grid, which can effectively support distributed greedy routing. In Kleinberg' s model, long links are added according to a harmonic probability distribution in the lattice distance space of a mesh

network [8]. Considering two nodes v and u on an $n \times n$ grid, the grid distance between nodes is defined as $d(v,u) = |k_v - k_u| + |l_v - l_u|$, where the coordinates of nodes v and u are (k_v, l_v) and (k_u, l_u) respectively. In addition to 4 direct neighbor links between adjacent nodes, each node adds one long link according to the harmonic probability distribution. The probability that node v links to node u is inversely proportional to the distance between them. That is, for a node u on the grid, the probability that v connects to it is $d(v,u)^{-r}/T$, where T is the normalization parameter of the harmonic probability distribution. Since probabilities are normalized across all network nodes, there must be one node linking to v. Queries are routed along the link whose grid distance to the target node is closest. When $r = 0$, the link length follows a uniform distribution, and the greedy routing hops between two points on the network are $O(n^{2/3})$. In [15], Kleinberg further extended the two-dimensional grid model to hierarchical tree network structures and general group structure models. Network topologies conforming to this model can be extended into small-world networks using Kleinberg' s model method, supporting deterministic greedy routing. Kleinberg' s model requires precise topology information T as the normalization parameter for the probability space of long links. Therefore, when actually constructing links, it is necessary to traverse the entire network or know the exact network size. When $r = 2$, routing hops can achieve $O(\log^2 n)$.

Literature [5] analyzed the routing efficiency of Kleinberg's small-world model on ring topologies and extended it to more general network structures. When the network topology has the property of distance-maintaining epimorphism with ring networks, Kleinberg' s small-world model can be used to add long links supporting deterministic greedy routing. Literature [21] verified that $O(\log^2 n)$ is the tight bound for the routing efficiency of Kleinberg' s small-world model and generalized it to d-dimensional mesh topologies. When $d = 1$, adding one long link can achieve $O(\log^2 n)$ query efficiency in d-dimensional meshes. In [23], the authors proposed a general model defining a class of network topologies that can be transformed to have the characteristics of Kleinberg' s small-world model. In [9], the authors used neighbor-of-neighbor information to improve routing efficiency to $O(\log n/d)$ in d-dimensional mesh networks. The Symphony protocol uses Kleinberg' s small-world model on ring networks to build long links through a local harmonic probability distribution generator [19]. The authors proved that adding k long links can achieve $O(\log^2 n/k)$ routing hops. Symphony' s local random function requires the actual network size as a normalization parameter, so it uses random sampling to estimate network size. The Mercury query protocol supports multi-attribute queries by building multiple Symphony rings [6], but adding long links requires estimating the current number of nodes in the network. Literature [4] and [30] add long links based on Kleinberg' s small-world model on d-dimensional Voronoi diagrams [11] to support multi-dimensional queries in different distance metric spaces. Long links are added using random walk methods. In [10], the authors studied how to build long links based on Kleinberg' s small-world model when node IDs are unevenly distributed in ring networks. In [7], the authors studied the char-

acteristics of topologies in routable small-world networks. In [8], the authors used random sampling methods to add long links on a tree-shaped peer-to-peer network topology to build a routable small-world network. Many unstructured peer-to-peer networks naturally have small-world network characteristics, such as Gnutella [22], but cannot support deterministic greedy routing. The method introduced in this paper targets the most general underlying network topologies in structured peer-to-peer networks, providing a general model framework for building routable structured small-world peer-to-peer networks, along with corresponding routing table construction methods and routing analysis methods.

3. Building Long Links in Ring Networks Based on Ring Distance

Let I be a linearly ordered ID space. Both network nodes and data select IDs from I. For unified discussion, we assume I is a continuous real interval $[0, H]$, where $H \geq 1$. n nodes randomly select a real number from I as their node ID. All nodes are arranged in increasing order of ID as $id_1, id_2, ..., id_n$, with neighboring nodes connected, and the node with the maximum ID connected to the node with the minimum ID, forming a ring underlying network. Data objects also select IDs from I. Node id_i is responsible for data objects whose IDs fall in the interval $(id_{i-1}, id_i]$. Looking up a target data t means finding the node responsible for that data. For two IDs x and y in I, the ring distance from x to y is defined as $d_{ring}(x, y) = \min\{(y - x) \mod H, (x - y) \mod H\}$. When processing a query request, a node selects the link in its routing table with the shortest ring distance to the target ID as the next hop. Each node maintains two short links pointing to its immediate predecessor and successor neighbors in the ring. To improve query speed, each node adds k long links. All long links are added in the same way. To add a long link, node id_i first generates a real number r in the interval $[0, H]$ according to a harmonic probability distribution as a LINK ID. It then finds the remote node responsible for this LINK ID in the network and adds it to the routing table as a long link.

Since we do not know the actual network size n, the LINK ID is actually generated in a virtual ring network S of size N, where $N > n$ is a predefined integer. The LINK ID r is generated by:

$$r = (id_i + H/e^x) \mod H \tag{1}$$

where x is a real number randomly selected from the interval $[0, N \ln b]$. It should be noted that the natural logarithm base e can be replaced by any other real number $b > 1$, then the interval for generating x becomes $[0, N \ln b]$, and it does not affect the distribution of r in the virtual ring network S. Node id_i finds the remote node j responsible for LINK ID r in the actual network and adds it to its routing table. The network generates k LINK IDs for each node to add multiple long links.

The parameters required for building long links include: node ID id_i, predefined ID space $[0, H]$, and predefined large integer N. These parameters are set by the application scenario, and all nodes use the same configuration without requiring additional dynamic global information. The only requirement is that node IDs are randomly selected and uniformly distributed in I. Compared with Symphony [15], this method does not need to estimate the actual network size. Using Symphony's proof method, we can prove that this method can construct a network structure approximating Kleinberg' s small-world model characteristics in ring networks.

First, consider the routing efficiency of the virtual ring network S with N virtual nodes. Let S denote the set of N virtual nodes arranged in increasing order of node IDs, $S = \{v_1, v_2, ..., v_N\}$. That is, the virtual nodes in S divide the ID space I into $N + 1$ equal-sized continuous intervals. Each virtual node adds k long links in the virtual ring network S according to equation (1). The routing efficiency has the following property.

Theorem 1. Each virtual node in S adds $k = O(\log N)$ long links, then the routing efficiency in S is $O(\log N)$ hops.

Proof: The detailed proof of this theorem can be found in [37].

If each node uses k LINK IDs to build long links in its routing table, when $N \geq n$ and nodes are uniformly distributed in I, the number of corresponding actual remote network nodes for these LINK IDs is $O(\log n)$, and the routing hops in the actual network are also $O(\log n)$.

Theorem 2. In a ring network with n nodes, each node uses $k = O(\log N)$ LINK IDs to add long links. If $N \geq n$ and nodes are uniformly distributed in the ID space, then the number of different remote nodes corresponding to the k LINK IDs is $O(\log n)$ with high probability, and the routing hops are also $O(\log n)$. Here "uniform distribution" means the ratio between maximum distance and average distance is bounded by a constant.

Assume n uniformly distributed network nodes in I are arranged in increasing order of node IDs as $id_1, id_2, ..., id_n$. Obviously, the n nodes divide the virtual ring network S into $n + 1$ intervals, and the expected number of virtual nodes each node is responsible for is N/n.

Arrange node id_i' s LINK IDs $r_1, r_2, ..., r_k$ in increasing order. Then the j-th LINK ID r_j can be expected to span 2^j virtual nodes from node id_i in the virtual ring network S. Thus, the first $k - \log_2 n$ LINK IDs of id_i can be expected to fall under the jurisdiction of id_i' s direct successor neighbor node id_{i+1} (see Figure 1(a)). The remaining $m = \log_2 n$ LINK IDs correspond to at most $\log_2 n$ network nodes. Since actual nodes are randomly selected, the number of virtual nodes they govern will not be exactly N/n. In intervals where the number of virtual IDs on virtual ring S is close to N/n, the maximum number of actual nodes will not exceed $O(\log n)$ with high probability. There may be very few intervals where the number of actual nodes exceeds $\log n$. This is equivalent to the classic

model of randomly throwing n balls into $n+1$ bins, where the maximum number of balls in any bin does not exceed $O(\log n)$ with high probability. Therefore, in general, k LINK IDs actually correspond to $O(\log n)$ network nodes.

If $N < n$, the n actual network nodes are divided by the N virtual nodes in S into N intervals, with each virtual node responsible for n/N actual network nodes. $\log N$ long links in the virtual network, i.e., LINK IDs, correspond to $\log N$ different actual network nodes. The query first goes through $\log N$ steps in S to reach the virtual node responsible for the target actual network node. After that, long links no longer work, and the query can only be forwarded along direct neighbor nodes in the actual network, requiring $O(n/N)$ steps. The total routing hops are $O(\log N + n/N)$. Figure 1(b) shows the routing scenario when $N < n$. Therefore, we need to choose a large integer N to ensure $N > n$.

Using the analysis method of Theorem 1, we can derive Lemma 1. This lemma gives the probability that a long link from a given node connects to another given node. Lemma 1 will be used in the virtual ring model introduced later.

Lemma 1. In a uniformly distributed ring network with n nodes, the probability that a long link built from node v connects to node u at distance l from v is $\Theta(1/(l \log n))$.

4. Virtual Ring Model

Next, we further extend the method of building long links based on the small-world model in ring networks from the previous section, and add long links on general structured peer-to-peer network underlying topologies using this method to form routable small-world network topologies. Our approach is: for a given underlying topology and a distance metric, first map the underlying topology to a ring network, then examine whether we can support deterministic greedy routing by adding long links. If yes, we can add long links in the virtual ring network and then map them back to the actual underlying network topology to build the actual node long links. To facilitate the introduction of the virtual ring model, we present its basic formal notation:

Let $G = (I, d, g, f)$ be a structured peer-to-peer network with n nodes, where: - I: represents the node ID space, - $d(v, u)$: represents the distance between nodes v and u in I, - g: represents the connected undirected underlying topology network. In the underlying topology network, each node links to its nearest nodes in the node space I, - f_v: is the hop count distribution of node v, i.e., the distribution of distances from node v to other nodes in the underlying topology, represented by $f_v(l)$, whose function value is the number of nodes at hop distance l from node v.

Queries use deterministic greedy routing, i.e., selecting the neighbor node closest to the target node (distance in ID space) as the next hop in routing. We use $d_g(v, u)$ to represent the shortest number of hops required to reach the target node during routing on the underlying network g, called the hop distance. The

hop distance distribution satisfies: $\sum_{l=0}^{l_{max}} f_v(l) = n$, $f_v(0) = 1$, where l_{max} is the farthest hop distance from node v on the network. When nodes are uniformly distributed in the ID space, we can naturally obtain $f_v(l) = \Theta(l^{d-1})$, for any l, where d is the dimension of the underlying topology network. For example, on a ring network based on ring distance, $f_v(l) = \Theta(1)$, $l_{max} = \Theta(n)$. In a d-dimensional torus network based on Manhattan distance, $f_v(l) = \Theta(l^{d-1})$. It should be noted that in an underlying topology network, all nodes v have the same hop distribution function, i.e., $f_v(l) = f(l)$. If the network's $f(l) = O(l^{d-1})$, then $l_{max} = O(n^{1/d})$.

The virtual ring model mainly includes four properties characterizing the underlying topology network, and provides corresponding long-link construction methods for different network topologies.

Property 1: When network nodes uniformly partition the ID space I (the ratio between the largest interval and the average interval size is bounded by a constant), the distance $d(v, u)$ in I between two nodes v and u with the largest hop distance $d_g(v, u)$ is also the largest, and the deviation from the average value can be expected not to exceed the logarithmic level $O(\log n)$.

This property can be used to analyze whether the ID space I of an underlying topology network is uniformly partitioned. Uniformly partitioning the ID space can help better analyze the constructed structured peer-to-peer network. However, this does not mean that strict uniform partitioning is required. We will introduce non-uniform distribution cases later.

We first introduce the specific definition of the virtual ring. For a given node v, its virtual ring is a ring network abstracted from the underlying network. In this virtual ring network, the first node is v, while other nodes consist of virtual nodes. Define $S_v(l) = \{u | d_g(v, u) = l\}$ as the set of all network nodes at hop distance l from node v, i.e., nodes on a virtual ring. This set satisfies $|S_v(l)| = f(l)$. Then node v' s virtual ring can be defined as a triple: $vVR = (I_v, VS_v, d_{ring})$, where I_v is a real interval $[0, L_{max}]$ in node space I, L_{max} represents the maximum distance in node space I according to distance metric d, and set VS_v consists of l_{max} virtual nodes arranged in increasing order of distance from node v as: $S_v(1), S_v(2), ..., S_v(l_{max})$. d_{ring} is the ring distance on interval I_v. For later discussion, we define set $B_v(l) = \bigcup_{i=0}^{l} S_v(i)$, i.e., the set of all nodes whose hop distance from node u is less than l.

A virtual ring has l_{max} virtual nodes, but we don' t actually know the l_{max} value. The size of l_{max} depends on the actual number of network nodes. However, we can predefine L_{max}. According to Property 1, when nodes uniformly partition the ID space, virtual nodes $S_v(l)$ can also uniformly partition I_v. This ensures we can apply the method from the previous section to add long links on the virtual ring network. It should be noted that each actual network node has a corresponding virtual ring network. Since we previously assumed all nodes share the same distance distribution f_v, we will only discuss in one virtual ring later, using vVR to denote node v' s virtual ring network.

Greedy routing starting from node v in the underlying network g can be viewed as greedy routing on vVR. One hop in vVR may correspond to multiple hops in the actual network, because one hop in the actual network does not necessarily reduce the hop distance to the target node. Similarly, one hop between virtual nodes in the virtual network does not necessarily reduce the hop distance to the target node in the actual network. To analyze the network' s routability properties, we present Property 2:

Property 2: Consider any node v and a given target node $u \in g$. Let c_v be the set of direct neighbor nodes of v, and $w \in c_v$ be a node in the direct neighbor set closest to u. Then $d_g(w,u) \leq d_g(v,u)$. If $d_g(w,u) < d_g(v,u)$, then the maximum hop distance on the network is less than l_{max}.

Property 2 indicates that any underlying network topology can be divided into the above two cases. Therefore, for these two classes of underlying topologies, we consider their link construction methods and respectively define Class A and Class B underlying topologies and their corresponding properties.

Definition 1: Class A Underlying Topology. In this class of underlying network topology, given a node v, for a target node u, if $d_g(v,u) = l$, then there exists at least one node $w \in S_v(l-1)$ such that $d_g(w,u) = l-1$.

In this case, we can apply the ring network method from the previous section to add $k = O(\log L_{max})$ long links for each virtual node in vVR. That is, we can use an integer L_{max} to generate LINK IDs. Assuming node v adds a long link to virtual node $S_v(l)$, the routing hops in vVR can reach $O(\log L_{max})$. This long link actually corresponds to selecting one or several actual network nodes from set $S_v(l)$ to link (determined by the actual network topology and distance distribution). Each node v corresponds to its own vVR. The long link from $S_v(l)$ to $S_v(l-1)$ in vVR actually corresponds to the link added by node $u \in S_v(l-1)$ in its ring virtual network uVR to the virtual node at distance $l-1$. Therefore, although we say we add k long links for each virtual node in vVR, we actually only need to consider the first virtual node.

For a given target node u, the long link from v may not necessarily help reduce the distance from the query message to the target node. Assume a virtual link of length l randomly selects an actual node from $S_v(l)$. According to Theorem 1 and Theorem 2, a virtual link can halve the distance to the target node (in the virtual network). However, in the actual network, it does not necessarily halve the actual network distance. The following Property 3 explains under what conditions long links on the virtual ring can be effective in the actual network.

Property 3: In a Class A underlying network topology, for any two nodes v and u, let $h_c(l) = \frac{|S_v(l) \cap B_u(l/2)|}{|S_v(l)|}$. If it satisfies:

$$h_c(l) \geq c, \quad \forall l \in [1, l_{max}]$$

where $0 < c \leq 1$ is a real number independent of l, then by adding $O(\log L_{max})$

long links per node, $O(\log n)$ routing efficiency can be achieved (proof see [37]).

Property 3 not only provides a method to analyze whether a Class A underlying topology can be extended into a routable small-world peer-to-peer network by adding long links, but also provides a framework for long-link construction. The key is h_c. That is, if analysis proves that h_c is a sufficiently large real number independent of l (less than or equal to 1), then we can build a routable structured peer-to-peer network by adding long links. For example, in ring networks based on ring distance, $h_c = \Theta(1)$; in d-dimensional torus networks based on Manhattan distance, $h_c = \Theta(1)$. In De Bruijn graphs, h_c is not independent of l but decreases exponentially with l. Therefore, De Bruijn graphs do not satisfy Property 3. We cannot improve the network's routing efficiency by adding long links (in fact, De Bruijn graphs already achieve $O(\log n)$ routing). We will introduce in Section 5 how to use Property 3 to build long links in d-dimensional torus networks.

Definition 2: Class B Underlying Topology. In this class of underlying topology, given a pair of nodes u and v, all nodes $w \in S_v(l)$ satisfy $d_g(w,u) = d_g(v,u)$ (see the right side of Figure 2).

In Class B underlying topologies, direct neighbor nodes can no longer be used as the next hop for greedy routing. That is, we cannot reduce the hop distance by 1 by entering a direct neighbor node. However, if the underlying network topology satisfies the following Property 4, we can still build a routable network.

Property 4: Given a node v and a target node u, for any node $w \in S_v(l)$, if there always exists a node z such that $d_g(z,u) \leq d_g(v,u) - 1$ and $d(v,z) \leq d(v,w)$, then by adding long links from v to a node in $\bigcup_{i=1}^{l} S_v(i)$, $O(\log n)$ routing hops can be achieved. Here $\pi(l)$ is a transformation on the integer interval $[0, l_{max}]$. Note that in the right half of Figure 2, $\pi(l) = l - 1$.

Like Property 3, Property 4 shows what kind of Class B underlying topology can be transformed into a routable network. In such networks, there always exists a set of nodes whose distance to the target node is one less than v' s distance to the target node. Thus, we can add a long link from node v to a node in this set to support greedy routing. We will discuss in Section 6 how to use Property 4 to build long links on ring networks based on hierarchical tree distance to support greedy routing.

5. Building Long Links in d-dimensional Torus CAN Networks Based on Manhattan Distance

In a d-dimensional CAN [26] peer-to-peer network, each node is represented by a d-dimensional vector $\mathbf{x} = (x_1, x_2, ..., x_d)$, where x_i is generated from the real interval $[0, H]$, with $H \geq 1$. The first node is responsible for the entire d-dimensional Cartesian coordinate space $D = [0, H]^d$. Subsequently joining nodes select an existing node on the network and partition one dimension of the interval responsible by that node in half. The dimension to be partitioned

is selected in a round-robin fashion. That is, starting from the first dimension, a newly joining node partitions the interval of an existing node along the i-th coordinate axis in half, where $i = (s+1) \mod d$, and s is the dimension selected by the existing node in its last partition (or when partitioning other nodes). Data objects are also represented by d-dimensional vectors in D and stored in nodes responsible for those coordinate vectors. To uniformly partition the d-dimensional space, joining nodes first randomly generate an ID uniformly distributed in $[0, H]^d$. The node then locates v on the network through an existing node, and finally partitions the interval of the node responsible for that ID using the round-robin dimension partition method. The joining node uses the center point coordinates of its responsible interval as its node ID. Each node establishes short links with adjacent nodes in each dimension. Nodes at the boundary of a dimension in space D connect to boundary nodes at the other end of that dimension, forming a loop. When nodes randomly select IDs and join using the round-robin dimension partition method, they approximately form a d-dimensional torus topology. Each node in the CAN peer-to-peer network maintains $O(d)$ short links, and routing hops can reach $O(dn^{1/d})$.

5.1 Using Virtual Ring Model to Build Long Links

In the d-dimensional ID space D of CAN peer-to-peer networks, Manhattan distance is defined as $d_{man}(v,u) = \sum_{i=1}^{d} \min\{|x_i - y_i|, H - |x_i - y_i|\}$ to evaluate the distance between two points $\mathbf{x} = (x_1, x_2, ..., x_d)$ and $\mathbf{y} = (y_1, y_2, ..., y_d)$ in the space, where:

$$\min\{|x_i - y_i|, H - |x_i - y_i|\}$$

is defined as the coordinate distance in dimension i. Since the maximum coordinate distance in each dimension is $H/2$, the maximum Manhattan distance L_{max} is $d \cdot H/2$.

In node v' s neighbors, there are always some nodes that can help reduce the distance to the target node. Therefore, the d-dimensional torus CAN peer-to-peer network belongs to Class A underlying topology, and we can use the one-dimensional virtual ring method to add long links. Figure 3 shows an example of mapping a two-dimensional CAN network to node v' s virtual ring network vVR. The squares in Figure 3(a) represent nodes at distances l_1, l_2, and l_3 from node v. These nodes are mapped to virtual nodes $S_v(l_1)$, $S_v(l_2)$, and $S_v(l_3)$ in vVR (Figure 3(b)). In an n-node d-dimensional CAN, the expected value of $|S_v(l)|$ is $O(l^{d-1})$. Therefore, $l_{max} = O(n^{1/d})$, and there are $O(dn^{1/d})$ virtual nodes corresponding to vVR.

When building long links, node v first generates k real numbers $r_1, r_2, ..., r_k$ in the interval $[0, L_{max}]$ according to a harmonic distribution, i.e., generates long links in vVR (the selection of constant k will be discussed in Section 5.2). Then for each r_i, it generates a vector point $\mathbf{y} = (y_1, y_2, ..., y_d)$ such that its

Manhattan distance from node v is r_i. $\mathbf{y}$ is used as the LINK ID to build v' s long link in the actual network. That is, node v looks up the node responsible for LINK ID $\mathbf{y}$ on CAN, adds it to its routing table as a long link, thereby mapping the links in vVR to the actual network.

First, generate in the interval $[0, L_{max}]$. The method used is similar to the LINK ID generation method in ring networks from Section 3, i.e.:

$$r_i = L_{max}/e^{x_i} \tag{2}$$

where x_i is a real number randomly selected from the interval $[0, d \ln H]$.

Equation (2) differs from equation (1) in that we do not directly incorporate node v' s ID, because node IDs are vectors that cannot be simply added with numerical distances. r_i is merely the length of the long link, i.e., the distance from the remote node to v.

Given a distance r_i, there can be multiple candidate LINK IDs $\mathbf{y}$ in the actual network. We randomly select one satisfying the condition. To do this, we randomly generate a real vector $\mathbf{s} = (s_1, s_2, ..., s_d)$ such that the distance from point $\mathbf{s}$ to v is r_i (the sum value loops back within $[0, H]$). To generate s_j, we initialize $\mathbf{s} = \mathbf{0}$ and repeat the following steps to obtain s_j:

a. Randomly select a real number from I as s_j;
b. Randomly assign a sign, positive or negative, to s_j with probability 1/2;
c. If $\sum_{i=1}^{j} |s_i| \geq r_i$, the program ends (all s_j have been obtained);

Finally, node v looks up the remote node responsible for $\mathbf{y}$ on the network and adds it to its routing table.

Routing must consider the size of the interval each node is responsible for in each dimension. Let $Z_v = ([z_1^-, z_1^+], [z_2^-, z_2^+], ..., [z_d^-, z_d^+])$ denote the d-dimensional interval currently responsible by node v, where $[z_i^-, z_i^+]$ is the one-dimensional interval occupied by v in dimension i. Define the interval Manhattan distance from v to target $\mathbf{y}$ as:

$$d_{interval}(Z_v, \mathbf{y}) = \sum_{i=1}^{d} d_{interval}(z_i, y_i)$$

where $d_{interval}(z_i, y_i)$ is the interval distance on coordinate axis i. If $y_i \in [z_i^-, z_i^+]$, then $d_{interval}(z_i, y_i) = 0$; otherwise, $d_{interval}(z_i, y_i) = \min\{|y_i - z_i^-|, |y_i - z_i^+|\}$. When $d_{interval}(Z_v, \mathbf{t}) = 0$, it means target point $\mathbf{t}$ falls within node v' s interval. In each hop of routing, the node selects from its routing table links the link with the shortest interval Manhattan distance to the target node as the next hop, until the interval Manhattan distance becomes zero, i.e., the target node is found.

5.2 Routing Efficiency of CAN Long Links

Different numbers of LINK IDs generate different routing table sizes and routing hops. At the same time, the dimension d of the space partitioned by CAN also affects the routing table and routing efficiency. When $k = O(\log N)$, each node will add $O(\log n)$ different remote nodes as long links, and routing hops can reach $O(\log n)$. We can derive Theorem 3 from Lemma 1 and the topological properties of the grid, with the help of Property 3. The key is the probability that a long link halves the distance. Figure 4 shows the basic proof idea.

Theorem 3. Assume on a d-dimensional CAN, n nodes uniformly partition the d-dimensional ID space using the round-robin dimension partition method, and each node uses k LINK IDs to build long links. Let N be a large integer constant, and $N^d \geq n$. Then the expected number of different long links added per node is $O(k)$. The expected routing hops are $O(\log n)$, and with high probability the maximum greedy routing hops can be $O(\log n)$.

Theorem 3 shows that we need to ensure $N^d \geq n$ to obtain an effective routing table. Assume we select a large integer N such that $N^d \geq Mn > n$, then $k = O(\log N) = O(\log n)$. Therefore, we set $k = O(\log N)$ LINK IDs to add long links.

5.3 Building Long Links in High-dimensional CAN Networks

When d increases, using 2^d LINK IDs to build long links is almost impossible. Fortunately, in most cases we can use $O(\log n)$ LINK IDs to add long links, and routing hops can reach $O(\log n)$. Moreover, when $d = O(\log n)$, CAN can achieve $O(\log n)$ routing hops without adding long links.

Lemma 2. If the expected longest path of an n-node d_1-dimensional CAN underlying network g_1 is smaller than that of an n-node d_2-dimensional CAN underlying network g_2, where $d_1 < d_2$ and both networks are built using the round-robin dimension partition method, then after adding links based on the virtual ring model to g_1 and g_2 respectively using the same number of LINK IDs, the expected longest path of g_1 will still not be greater than that of g_2 (see example in Figure 5 [Figure 5: see original paper]).

Lemma 2 shows that we can achieve the same routing efficiency in high-dimensional CAN networks with fewer long links as in low-dimensional CAN networks, thus avoiding the problem of exponential increase in long links. However, this depends on the number of nodes and the dimension of the network. The longest path length of a d-dimensional CAN is $O(dn^{1/d})$, so the path length of a d_1-dimensional CAN network is smaller than that of a d_2-dimensional CAN with the same number of nodes. Considering $d_2 = d_1 + 1$, there exists a specific n such that when $n > n_t$, the path length of the d_1-dimensional network is smaller than that of the d_2-dimensional network. It can be shown that n_t is relatively small. Figure 6 Figure 6: see original paper shows the curve of function $t(n, d)$ changing with d. Therefore, when

the network node count n is sufficiently large, using $O(\log n)$ LINK IDs can achieve routing efficiency no worse than the two-dimensional case.

Theorem 4. When using the round-robin dimension partition method to partition d-dimensional space and $d > \log n$, each node has at most $2\log n$ direct neighbors. Each node uses $O(\log\log n)$ LINK IDs to add long links ($N^d \geq n$), and routing hops are $O(\log n)$.

When $d > \log n$, network nodes actually have at most $2\log n$ direct neighbor nodes. At this point, the maximum number of virtual nodes $L_{max} = 2\log n$. That is, each node has one neighbor node in each of its first $2\log n$ dimensions. Without long links, routing can only proceed along direct neighbor nodes. A query requires at most one hop in each dimension, $O(\log n)$ hops to reach the target node. Using $O(\log\log n)$ LINK IDs to build long links, since there are only 2 long links of length r in vVR, the probability of helping reduce one hop to the target distance is $1/2^l$. $O(\log\log n)$ long links can help reduce at most a constant number of routing hops. Therefore, routing hops remain $O(\log n)$.

In summary, in high-dimensional CAN networks, we don' t need to use 2^d LINK IDs. Each node builds $O(\log n)$ links in total, and queries can achieve $O(\log n)$ routing efficiency. When $d > \log n$, the d-dimensional CAN network topology can be viewed as a CAN network structure with dimension $2\log n$. In this case, the function $t_f(d)$ can be transformed into a function of n by substituting $2\log n$ for d. Figure 6(b) shows the curve of function $t_f(n)$. Even when n is large, $t_f(n)$ remains small. Therefore, in most high-dimensional networks, we can use $O(\log n)$ LINK IDs to add long links to achieve routing efficiency that grows logarithmically with network scale.

It should be noted that although $O(\log n)$ routing efficiency can be achieved in networks with dimension $d = O(\log n)$ without adding long links, it is still difficult to achieve $O(\log n)$ routing hops by adding more long links. Only when using $O(\log n)$ links, with each node linking to almost all other nodes on the network, can $O(\log n)$ hops be achieved. Moreover, when $d > \log n$, each node is responsible for a larger interval, i.e., responsible for all data on dimensions that have not been split, causing the actual search space to become larger. Literature [31] proposed a Rolling Index data structure to solve the problem of data retrieval in high-dimensional space. Its main method is to merge multiple dimensional coordinates into one coordinate, thereby reducing space dimensionality. eCAN [12] improves routing efficiency on CAN networks by building long links, achieving $O(\log n)$ hops [34]. eCAN dynamically tracks the node splitting process to build and adjust long links. In [35], the authors use the node splitting tree path formed during CAN construction as node IDs and data IDs, forming a ring network on which they build a Skip Graph index structure to support range location queries for data. To set node IDs, nodes need to record the dimensions they have split and the dimensional coordinates of split points. Since node departure and joining change the splitting tree structure, causing corresponding node ID changes, the dimensional coordinates of split points are no longer accurate, and the relationship determining target data IDs and node

IDs also changes due to splitting tree changes. If network topology maintenance cannot accurately reflect these changes, routing cannot be guaranteed to always converge. The process of building long links on CAN based on the virtual ring model can be performed after the entire CAN topology is completed, or can be added along with CAN construction, and is easy to maintain. Each node records incoming links, and when a node leaves, it moves the incoming links to the corresponding neighbor nodes, ensuring the correctness of routing tables without recalculating LINK IDs to generate long links.

Under uniform partitioning of d-dimensional space, long links added based on the virtual ring model can effectively support routing. Uniform partitioning means each node is responsible for intervals of the same size in space. For example, in an extreme case where all nodes only split along a fixed dimensional coordinate, forming a ring network topology, the method for adding long links does not need to change, and the routing table built through the virtual ring model naturally degenerates to long links in ring networks. The virtual ring model is also suitable for other regularly partitioned multi-dimensional space underlying network topologies, such as Delaunay graphs.

When supporting range queries, distributed hash tables cannot be used to hash data into multi-dimensional ID space; data can only be published according to its original values. At this time, the load distribution in multi-dimensional space will be very uneven. Static load balancing methods can be used to uniformly partition the data space, or dynamic load balancing methods can be used to reduce the load of overloaded nodes during runtime. Static load balancing means that when the data distribution is known in advance, the interval size is partitioned according to the data distribution, and the data space is evenly divided while building the network to achieve load balancing. At the same time, the network topology can remain regular, and the virtual ring model can be applied to add long links. In dynamic cases, the global data distribution usually cannot be obtained in advance. At this time, nodes or data can be moved according to local load distribution information to reduce the load of overloaded nodes. When moving nodes, nodes with lighter load on the network can be selected to move to overloaded intervals to share the load of overloaded nodes. However, moving nodes in multi-dimensional space will cause the node density in overloaded intervals to be greater than that in intervals with smaller load, resulting in the underlying network topology no longer being regular and routing efficiency decreasing. At this time, although long links added using the virtual ring model can still guarantee routing efficiency to a certain extent, they will cause some nodes responsible for larger spaces (but with less actual load) to have too many incoming links. At the same time, in extreme cases, such as when data continuously accumulates in a very small interval, routing efficiency will decrease. Moving data can also achieve load balancing. In multi-dimensional CAN, load balancing can be achieved by changing node boundaries and only moving data between direct neighbor nodes, without storing additional data pointers, while keeping the original network topology unchanged. However, the convergence speed of load balancing only between neighbor nodes is slower,

usually related to the network diameter or expansion ratio. In high-dimensional CAN, the network diameter reaches $O(\log n)$, so load balancing can converge quickly. In two-dimensional networks, an $O(\sqrt{n})$ network diameter can also be achieved.

6. Building Long Links in Ring Networks with Hierarchical Tree Distance

Kleinberg extended the small-world model on two-dimensional grid networks to networks with hierarchical tree distances in [15], building long links based on harmonic distribution to support deterministic greedy routing on networks based on hierarchical tree distances. In networks based on hierarchical tree distances, node IDs are path IDs from the root node to leaf nodes, represented by a b-ary string. The distance between nodes is prefix distance, i.e., the length of the shared prefix of two node IDs. The probability of node v linking to a remote node u is inversely proportional to their prefix distance. Each node independently selects $O(\log n)$ remote nodes as long links based on harmonic distribution in the network, and routing hops can reach $O(\log n)$. Similar to two-dimensional grid networks, Kleinberg' s hierarchical tree network model still requires knowledge of the entire network size to determine the normalization parameter of the harmonic distribution. Below we briefly introduce how to use the virtual ring model to analyze and build long links on ring networks based on hierarchical tree distances.

In networks based on hierarchical tree distances, node IDs are represented by b-ary strings of length h, and the tree distance between two nodes v and u is defined as the length of the shared prefix of their node IDs. For simplicity, we assume $N = b^h$. Like in ring networks and CAN networks, we need to ensure N is greater than the actual number of network nodes $O(\log n)$. Therefore, $h = O(\log_b n)$. When n nodes randomly select node IDs, the maximum node distance in the actual network is $l_{max} = O(\log_b n)$. We use the virtual ring model to map the hierarchical tree network to the virtual ring network vVR. However, hierarchical tree networks belong to Class B virtual rings (their virtual ring construction method is shown in Figure 7 [Figure 7: see original paper]). In a hierarchical tree network, given a query initiated from node v with target node u at distance $l = td(v, u)$, we know [15]:

1. A long link can reduce the distance to the target node by at least 1 step if and only if it points to a subtree of height l that contains u, where $l \leq h - 1$;
2. Except for the subtree of height l where v is located, there are $b - 1$ such subtrees, and only one of them contains u;
3. If each hop in the network can reduce the distance to the target node by one step, after h hops, the distance to the target node is at most 1. If nodes in subtrees of height 1 are linked together, when the distance to the target node is reduced to 1 (i.e., falling into a subtree of height 1), at most b more hops are needed to locate the target node.

The hierarchical tree network model belongs to Class B virtual rings. From (1) we can get $\pi(l) = l - 1$. Assume a query in vVR starts from node v_1 (i.e., the starting node v) with target virtual node v_k, at distance l_{max} from v_1, requiring traversal of $\log_b n$ virtual nodes $v_1, v_2, ..., v_k$. Let $r(v)$ be a long link of node v. If $r(v) = p_i$, then at the i-th hop of the query, the distance to the target node is $d_g(p_i, u) = l - i + 1$. To ensure the routing hops in the actual network are $\log_b n$, a simple method is to set $p_i = v_i$ for all i. That is, node v at distance l from target u should have a long link pointing to virtual node $S_v(l)$ in vVR. Since the possible distances between v and target node u are $0, 1, ..., h$, node v needs to add $\log_b n$ long links in vVR. The i-th longest link is at distance i from v. The virtual node $S_v(i)$ in vVR corresponds to the root nodes of $b - 1$ subtrees of height i in the actual network, and target node u may fall into any of these subtrees. Therefore, a long link in the virtual network needs to correspond to the root nodes of each subtree, i.e., correspond to $(b - 1)$ long links of length i in the actual network.

The method for obtaining routing tables in hierarchical tree distance space using the virtual ring model actually has the same characteristics as Tapestry and Pastry. Both also use prefix-length-based routing methods, and routing tables are also built based on the prefix length of node IDs, thus having the same structure as routing tables in hierarchical tree distance space. Kleinberg' s tree network model does not emphasize what topology the underlying network uses to connect all nodes. Therefore, $O(\log n)$ links need to be added to ensure network connectivity.

We specifically designed and verified a routing table structure based on hierarchical tree distance on a ring network to support greedy routing based on hierarchical tree distance. For simplicity, in an n-node network, we use random characters of length h in a b-ary string as node IDs, i.e., ensuring $N = b^h \geq n$. The strings are linearly ordered by one-dimensional string dictionary order to form a ring network g. Node id_i is responsible for nodes whose IDs fall in the interval $(id_{i-1}, id_i]$ according to dictionary order. The distance between two points is the length of the shared prefix of IDs. Assuming node v' s ID is a b-ary string id_i of length h, for $l = 1, 2, ..., h$, each node randomly generates $(b - 1)$ LINK IDs of length h at distance l from v. A LINK ID at distance l shares a prefix of length l with id_i, with remaining characters randomly generated. After generating all $(h(b - 1))$ LINK IDs, node v looks up the nodes responsible for these LINK IDs on the ring network and adds them to its routing table as routing links. Since $h = O(\log_b n)$ and nodes randomly select IDs, the number of actual long links in the routing table is $O(\log n)$.

The query tries its best to reduce the distance between the predecessor node and the target node on the network. Since nodes are actually responsible for an interval of IDs, routing must consider the size of the interval the node is responsible for to ensure routing convergence. Given a query issued from node v with target node t, let rid be a long link in node v' s routing table. The distance between the node pointed to by the long link and node v is defined

as $d_{id}(rid, t)$, where id_{rid-1} is the ID of node rid' s direct predecessor. During routing, node v selects the long link rid with the smallest distance $d_{id}(rid, t)$ as the next hop node. Since $d_{id}(id_i, t) \geq d_{id}(id_{i+1}, t)$, for $d_{id}(id_i, t) > d_{id}(w, t)$, routing convergence can be guaranteed. Similar to Pastry and Tapestry, when nodes uniformly partition the ID space, routing hops are $O(\log n)$.

7. Simulation Experiments

To verify the network routing efficiency after adding long links using the virtual ring model, we conducted simulation experiments on ring networks with ring distance, CAN networks with d-dimensional Manhattan distance, and ring networks with hierarchical tree distance. The main experimental metrics are average routing hops, maximum routing hops, and average routing table size. Unless otherwise specified, routing table size only refers to the number of long links. To evaluate routing efficiency, n queries are issued from a fixed node on an n-node network, with each node on the network accessed once to collect statistics on average and maximum routing hops.

First, we tested routing characteristics in ring networks. We compared the routing hop distributions on two 1024-node ring networks. We strictly followed Kleinberg' s model method to add one long link per node on a ring network (represented by Kleinberg in Figure 8 [Figure 8: see original paper]). That is, each node could obtain the exact normalization parameter T and add each long link with probability proportional to the inverse of its distance from the node (after normalization) by traversing the entire network. Then, we used the virtual ring model method to add long links on the ring network using k LINK IDs to build a structured small-world network. From Figure 8, we can see that nodes in the pure Kleinberg small-world network are regularly distributed, thus having relatively short average routing hops. In the approximately constructed small-world ring network, nodes are randomly distributed and do not strictly uniformly partition the network ID space.

Figure 9 [Figure 9: see original paper] shows the relationship between the predefined integer N and routing table size and routing efficiency in ring networks. When $N = 162$, both routing table size and average routing hops can reach $O(\log n)$. In Figure 9(b), we set $N = 102$, smaller than the actual network size. It can be seen that the routing table size remains stable at this time, not increasing with the number of network nodes, while routing hops increase faster. The experiments show that selecting a large integer N to generate LINK IDs can ensure the effectiveness of long links in routing tables.

Furthermore, we tested the routing efficiency of CAN after adding long links based on the virtual ring model. For convenience, we call the CAN network after adding long links using the virtual ring model SCAN.

Figure 10 [Figure 10: see original paper] shows the routing hops and routing table size on SCAN networks with dimensions 2, 3, and 4. To make routing hops reach $O(\log n)$, each node uses $k = O(\log N)$ LINK IDs to build long links.

All network routing table constructions use $M = 202$. For easy comparison, we provide theoretical upper bounds for maximum routing hops, average routing hops, and average routing table size in the figures. Since all theoretical upper bounds are $O(\log n)$, we use $C \log N$ to represent them, where C represents integer values 1, 2, 8, 32, and 128. d_{max_qr} represents maximum routing hops, d_{avg_qr} represents average routing hops, and d_{avg_rt} represents average routing table size, where d is the number 2, 3, and 4. Figure 10 shows that the upper bound for maximum routing hops is $O(\log n)$, the upper bound for average routing table size is $O(\log n)$, and the upper bound for average routing hops is also $O(\log n)$, thus verifying the conclusion of Theorem 3.

Next, we verify the conclusion of Lemma 2, that using a smaller number of LINK IDs can achieve $O(\log n)$ routing hops. In the verification, we tested the maximum and average routing hops in SCAN networks with dimensions 2, 3, 4, and 5, with node count n ranging from 2^6 to 2^{14}. First, we used $k = 8 \log_2 N$ LINK IDs to build long links, with $M = 202$. Figure 11 Figure 11: see original paper shows that SCAN networks with dimensions 2, 3, 4, and 5 can achieve almost the same average routing hops and maximum routing hops, with $O(\log n)$ as the upper bound. When using $k = 4 \log_2 N$ LINK IDs to build long links, the routing hops of all networks are still very close. When using $k = 1$ LINK ID, Figure 11(c) shows that the routing hops of high-dimensional networks are smaller than those of low-dimensional networks. The experiments show that we can use $O(\log n)$ LINK IDs to add long links in high-dimensional SCAN networks to achieve $O(\log n)$ routing efficiency. When $d = O(\log n)$, CAN can achieve $O(\log n)$ routing hops without adding long links.

To further verify that using a limited number of LINK IDs to build long links can achieve effective routing hops, we tested the trend of routing table size and routing hops with node count n on CAN and SCAN networks with dimension $d = \log_2 n$. Figure 12 [Figure 12: see original paper] shows the average routing table size of CAN networks with $N = 202$ and dimension d from 2 to 20. Figure 12(b) shows that when $d > 10$, the average routing table size of CAN networks no longer increases (CAN routing table size is represented by direct neighbor count). When using $O(\log \log n)$ LINK IDs to add long links, the average number of long links (represented by SCAN_{avg} in Figure 12(a)) has an upper bound of $O(\log \log n)$ (represented by $4 \log \log N$ in Figure 12(a)). When $n = 1024$, no long links are added (represented by LOG in the figure), where $N = 202$, $d = 2, 3, ...$, average routing hops (SCAN_{avg} in Figure 12(b)) and maximum routing hops (SCAN_{max} in Figure 12(b)) are both bounded by $O(\log n)$ (represented by LOG in Figure 12(b)). Using $O(\log \log n)$ LINK IDs to build long links, the maximum routing hops (SCAN_{max}) decrease. The above experiments show that using $O(\log \log n)$ LINK IDs, each node maintains $O(\log n)$ connections (including long and short links), and queries can achieve $O(\log n)$ routing efficiency.

Finally, we simulated and tested the effect of adding long links in hierarchical tree distance space using the virtual ring model on a ring network. Figure 13

[Figure 13: see original paper] shows the trend of routing table size and routing hops with network node count. In Figure 13(a), 2-20_qr indicates that the network' s ID space consists of b-ary strings of length $h = 20$, $b = 2$. 10-6_qr indicates $h = 6$, $b = 10$. It can be seen that the routing hops have an upper bound of $O(\log n)$ (LOG10 and LOG2 in Figure 13 represent theoretical upper bounds $O(\log_{10} n)$ and $O(\log_2 n)$ respectively). Figure 13(b) shows the routing table sizes in two cases: $b = 2$, $h = 20$ (represented by 2-20_rt) and $b = 10$, $h = 6$ (represented by 10-6_rt). It can be seen that both have an upper bound of $O(\log n)$.

8. Conclusion

The virtual ring model, for specific distance metrics and underlying network topologies, builds long links on the basis of structured peer-to-peer networks based on Kleinberg' s small-world model, forming structured small-world networks that support deterministic greedy routing. Applying this method to d-dimensional CAN with Manhattan distance, using $O(\log n)$ LINK IDs, we can achieve $O(\log n)$ routing hops. In high-dimensional CAN, we can use $O(\log \log n)$ LINK IDs to achieve $O(\log n)$ routing efficiency. In hierarchical tree distance space, we use the virtual ring model to analyze and design a ring topology network that can effectively support greedy routing based on hierarchical tree distance and prefix distance. Using $O(\log n)$ LINK IDs, each node adds $O(\log n)$ links, and we can achieve $O(\log n)$ routing hops. The virtual ring model considers both network topology and measurement methods in data space, has good generality and practicality, and can serve as a general framework for routable small-world models, as well as a design and analysis framework for structured peer-to-peer networks, with good operability and scalability.

This work was funded by the 973 "Semantic Grid and Knowledge Grid" project (No. 2003CB317000).

References:

[1] K. Aberer, L. O. Alima, A. Ghodsi, S. Girdzijauskas, M. Hauswirth, S. Haridi, "The Essence of P2P: A Reference Architecture for Overlay Networks" , Proc. 5th IEEE International Conference on Peer-to-Peer Computing, Konstanz, Germany, August 2005

[2] R. Albert and A.-L.Barabasi, "Statistical Mechanics of Complex Networks," Reviews of Modern Physics, vol. 74, no. 47, 2002

[3] J. Aspnes and G. Shah, "Skip Graphs" , Proc. SODA, 2003

[4] F. Banaei-Kashani, C. Shahabi, "SWAM: A Family of Access Methods for Similarity-Search in Peer-to-Peer Data Networks" , Proc. ACM CIKM 2004, pp: 304 - 313, 2004

[5] L. Barrire, P. Fraigniaud, E. Kranakis, and D. Krizanc, "Efficient Routing

in Networks with Long Contacts" , Proc. 15th International Symposium on Distributed Computing, pp. 270-284, 2001

[6] A. Bharambe, M. Agrawal, and S. Seshan, "Mercury: Supporting Scalable Multi-Attribute Range Queries" , Proc. SIGCOMM, 2004

[7] P. Duchon, N. Hanusse, E. Lebhar, and N. Schabanel, "Could Any Graph Be Turned Into A Small World?" , Theoretical Computer Science, Vo. 355, No.1, pp.96–103, 2006

[8] P. Duchon, N. Hanusse, E. Lebhar and N. Schabanel, "Towards Small World Emergence" , Proc. the Eighteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures, pp.225-232, New York, NY, 2006

[9] P. Fraigniaud, C.l Gavoille, C. Paul, "Eclecticism Shrinks Even Small Worlds" , Proc. 23rd ACM PODC, pp.169-178, 2004

[10] S. Girdzijauskas, A. Datta, K. Aberer, "On Small World Graphs in Non-uniformly Distributed Key Spaces", Proc. The 1st IEEE International Workshop on Networking Meets Databases (NetDB), Tokyo, Japan, 2005

[11] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, I. Stoica, "The Impact of DHT Routing Geometry on Resilience and Proximity" , Proc. ACM SIGCOMM 2003, Karlsruhe, Germany,

[12] N. Harvey, M. Jones, S. Saroiu, M. Theimer, and A. Wolman, "Skip-Net: A Scalable Overlay Network with Practical Locality Properties" , Proc. Fourth USENIX Symposium on Internet Technologies and Systems, pp. 113-126, Mar. 2003

[13] M. F. Kaashoek and D. R. Karger, "Koorde: A Simple Degree-Optimal Distributed Hash Table" , F. Kaashoek and I. Stoica (Eds.): IPTPS 2003, LNCS 2735, pp. 98 -107, 2003

[14] J. Kleinberg, "The Small-World Phenomenon: An Algorithmic Perspective" , Proc. 32nd ACM STOC, pp. 163-170, 2000

[15] J. Kleinberg, "Small-World Phenomena and the Dynamics of Information" , Advances in Neural Information Processing Systems (NIPS) 14, 2001

[16] M. Li, W. C. Lee, A. Sivasubramaniam, "Semantic Small World: An Overlay Network for Peer-to-Peer Search" , Proc. 12th IEEE International Conference on (ICNP' 04), pp. 228-238, 2004

[17] D. Loguinov, J. Casas, X. Wang, "Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience" , IEEE/ACM Transactions on Networking, Vol. 13, Issue 5, pp.1107 - 1120, October 2005

[18] S. Milgram, "The Small World Problem" , Psychology Today, vol.1, no.61, 1967

[19] G. Manku, M. Bawa, and P. Raghavan, "Symphony: Distributed Hashing in a Small World" , Proc. the 4th USENIX Symposium on Internet Technologies

and Systems, 2003

[20] D. Malkhi, M. Naor, D. Ratajczak, "Viceroy: a scalable and dynamic emulation of the butterfly" , Proc. 21st ACM PODC, pp. 183-192, 2002

[21] C. Martel and V. Nguyen, "Analyzing Kleinberg' s (and other) Small-world Models" , Proc. 23rd ACM Symp. on Princ. of Dist. Comp. PODC' 04, pp. 179-188, 2004

[22] R. Matei, A. Iamnitchi, P. Foster, "Mapping the Gnutella Network" , IEEE Internet Computing, Vol. 6, No. 1, pp.50 - 57, Jan.-Feb. 2002

[23] V. Nguyen, C. Martel, "Analyzing and Characterizing Small-world Graphs", Proc. 16th ACM-SIAM Symposium on Discrete algorithms, pp: 311 - 320, 2005

[24] C. G. Plaxton, R. Rajaraman, and A.W. Richa, "Accessing Nearby Copies of Replicated Objects in A Distributed Environment", Proc. ACM SPAA, pp. 311-320, Jun. 1997

[25] C. Qu, W. Nejdl, and M. Kriesell, "Cayley DHTs-A Group-Theoretic Framework for Analyzing DHTs Based on CayleyGraphs" , Proc. International Symposium on Parallel and Distributed Processing and Applications (ISPA), 2004

[26] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network" , Proc. SIGCOMM 2001, Aug. 2001

[27] A. Rowstron, and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-Peer Systems", Proc. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pp. 329-350, 2001

[28] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications" , Proc. ACM SIGCOMM, pp. 149 -160, Aug.

[29] A. Spognardi and R. Di Pietro, "A Formal Framework for the Performance Analysis of P2P Networks Protocols" , Proc. IPDPS 2006, pp.8, 2006

[30] M. Steiner and E. Biersack. "Shortcuts in a Virtual World" , in Proceedings of CoNext 2006, 2006

[31] C. Tang, Z. Xu, S. Dwarkadas, "Peer-to-peer Information Retrieval Using Self-Organizing Semantic Overlay Networks" , Proc. SIGCOMM 2003, pp.175-186, 2003

[32] D. Watts and S. Strogatz, "Collective Dynamics of small-world networks" , Nature 393, pp. 440-2 ,1998

[33] J. Xu, A. Kumar, X. Yu, "On the Fundamental Tradeoffs Between Routing Table Size and Network Diameter in Peer-to-Peer Networks" , IEEE Journal on Selected Areas in Communications, Vol. 22, Issue: 1, pp: 151- 163, Jan. 2004

[34] Z. Xu and Z. Zhang, "Building Low-maintenance Expressways for P2P Systems," Hewlett-Packard Labs, Palo Alto, CA, Tech. Rep. HPL-2002-41, 2002

[35] C. Zhang, A. Krishnamurthy, and R.Y. Wang, "SkipIndex: Towards a Scalable Peer-to-Peer Index Service for High Dimensional Data" . Technical Report (TR-703-04), Princeton University, 2004

[36] B. Y. Zhao, H. Ling, J. Stribling, S. C. Rhea, A. D. Joseph, J. D. Kubiatowicz„ "Tapestry: A Resilient Global-Scale Overlay for Service Deployment" , IEEE Journal on Selected Areas in Communications, vol.22, no.1, pp. 41 -53, 2004

[37] H. Zhuge, X. Sun, "A Virtual Ring Method for Building Small-world Structured P2P Overlay Networks" , IEEE Trans. On Knowledge and Data Engineering, vol.20, no.12, pp. 1712-1725. 2008.

Author Biographies:

Zhuge Hai: Researcher, Ph.D., Institute of Computing Technology, Chinese Academy of Sciences

Sun Xiaoping: Assistant Researcher, Ph.D., Institute of Computing Technology, Chinese Academy of Sciences

This paper has been published in English in IEEE Trans. On Knowledge and Data Engineering, vol.20, no.12, pp. 1712-1725. 2008.

Note: Figure translations are in progress. See original paper for figures.