A Scalable Peer-to-Peer Network with Constant Degree*

Dongsheng Li[1], Xinxin Fang[2], Yijie Wang[1], Xicheng Lu[1], Kai Lu[1], and Nong Xiao[1]

[1]School of Computer,
National University of Defense Technology,
410073 Changsha, China
leedongsh@hotmail.com
[2]Department of Compute science and technology,
Tongji University,
200092 Shanghai, China
sabina_xin@hotmail.com

Abstract. Recently many distributed hash table (DHT) schemas have been proposed to build scalable peer-to-peer systems, in which degree and diameter are two important measures. In this paper, we propose *Fission*, a novel DHT-style peer-to-peer network, which is of constant degree and $O(\log N)$ diameter. Peers in Fission form an approximate Kautz topology and the "split large and merge small" policy is exploited to achieve load balance when peers join or depart. The performance of Fission is evaluated using both analysis and simulation. Formal proofs verify that the degree and the diameter of Fission are no more than $O(1)$ and $2*\log N$ respectively and the join or departure of one peer requires only $O(1)$ peers to change their state. Simulations show that the load balance characteristic of Fission is good and the average path length of Fission is no more than $\log N$.

1 Introduction

In recent years, peer-to-peer computing has emerged as a novel and popular computation model and attracted significant attentions from both industrial and academic fields [1,2]. The core component of many proposed peer-to-peer storage systems [3,4,5,6,7,8] is the distributed hash table (DHT) schema that uses a hash table-like interface to publish and lookup data objects. In DHT schemas, each peer bears responsibility for a certain portion of the key space and uses a routing table to forward the query for data object. To maintain the DHT, the responsibility is re-assigned between peers when peers join or depart.

Two important measures of DHT systems are degree, the size of routing table to be maintained on each peer, and diameter, the number of hops a query needs to travel in the worst case. In many existing DHT schemas, such as Chord [3], Tapestry [4] and Pastry [5], both the degree and the diameter tends to $\log N$. In this paper, we propose

* This work was supported by National Natural Science Foundation of China under the grant No. 69933030, 60203016 and 90104001, Excellent PHD Dissertation Foundation of China under the grant No. 200141 and National 863 High Technology Plan of China under the grant No. 2002AA131010.

X. Zhou et al. (Eds.): APPT 2003, LNCS 2834, pp. 414–424, 2003.

Fission, a novel DHT-style peer-to-peer network with constant degree and $O(\log N)$ diameter, which exploits Kautz topology and uses the "split large and merge small" policy for maintenance. Fission is inspired by CAN [6], which allows peers re-positioning the space, and D2B [9], a DHT schema based on de Bruijn graphs. However, Fission can achieve better results than both. Proofs show that the degree and diameter of Fission is $O(1)$ and $O(\log N)$ respectively even in the worst case. Compared with Fission, CAN uses a d-dimensional torus topology with $2d$ degree, but the diameter of CAN is $O(dN^{1/d})$ and so it doesn't scale quite as well as Fission. D2B, which is based on de Bruijn graphs, provides constant expected degree and $O(\log N)$ diameter, but its degree could be at most $O(\log N)$ with high probability. That is to say, some peers in D2B will be likely to have $O(\log N)$ degree. Chord [3] uses a ring topology; Tapestry [4] and Pastry [5] utilize hypercube topology based on prefix-based routing. Their diameters are $O(\log N)$, but their degrees are $O(\log N)$ which is larger than that of Fission. Viceroy [10] is based on butterfly topology with constant expected degree and $O(\log N)$ diameter, but its degree and diameter could be $O(\log N)$ and $O(\log^2 N)$ respectively in the worst case.

The remainder of the paper is organized as follows. Section 2 introduces Kautz graph. Section 3 describes the design and properties of Fission. Section 4 presents the routing algorithm in Fission and proves its correctness. Section 5 evaluates the performance of Fission. Conclusions and future work is discussed in Section 6.

2 Kautz Graph

Fission exploits Kautz graph as its topology. Given two positive integers d and k, a Kautz graph [11,12], denoted by $K(d,k)$ is a directed graph with degree d and diameter k. The nodes of Kautz Graph are encoded with a string of length k where each symbol of the string is from the set Z={0,1,2,…,d} with the restriction that two consecutive symbols of the string are always different. That is to say, node u is encoded with $u_1u_2\ldots u_k$ where $u_i \neq u_{i+1}$, $1\le i\le k-1$, $u_i \in Z$. Obviously there are $N=d^k+d^{k-1}$ nodes in the $K(d,k)$ graph. Each node $u = u_1u_2\ldots u_k$ has d outgoing edges: for each $\alpha \in Z$ and $\alpha \neq u_k$, node u has one outgoing edge to node $v = u_2u_3\ldots u_k\alpha$ (denoted by $u \rightarrow v$), i.e., there is an edge from u to v iff v is a left-shifted version of u. Therefore, each node in $K(d,k)$ is of in-degree d and out-degree d. Figure 1 shows $K(2,3)$.

Routing in Kautz graph from node u to node v is accomplished by taking the string u and shifting in the symbols of v one at a time until the string u has been replaced by v. For instance, given two nodes $u = u_1u_2\ldots u_k$ and $v = v_1v_2\ldots v_k$ the routing path from u to v is

$$u = u_1u_2\ldots u_k \rightarrow u_2u_3\ldots u_kv_1 \rightarrow u_3u_4\ldots u_kv_1v_2 \rightarrow \ldots \rightarrow u_kv_1v_{2\ldots}v_{k-1} \rightarrow v_1v_{2\ldots}v_k \quad (1)$$

when $u_k \neq v_1$ or a path of length k-1 as below:

$$u = u_1u_2\ldots u_k \rightarrow u_2u_3\ldots u_kv_2 \rightarrow u_3u_4\ldots u_kv_2v_3 \rightarrow \ldots \rightarrow u_kv_{2\ldots}v_{k-1}v_k = v_1v_{2\ldots}v_k \quad (2)$$

Compared to de Brujn graph [13] with degree d and diameter k, which has d^k nodes, Kautz graph $K(d,k)$ has $(1+1/d)$ times the number of nodes $B(d,k)$, i.e., with the same total number of nodes N and the same value of d, the diameter of Kautz graph is smaller than that of de Brujn graph. Also it is known that there are obvious uneven loads of edges in a de Brujn graph. Considering any node u=ii…i, $0\le i\le d-1$, the edge

from ii…i to ii…iia for any 0≤a≤d-1 is used only when the node u is the source node, i.e., node u never forward any query. That is because if there is a node v=bii…i having an outgoing edge to u, it also has an outgoing edge to ii..iia. Further features [12] show that $K(d,k)$ is a better logical topology than $B(d,k)$, so Kautz graph is chose as the topology of the Fission.

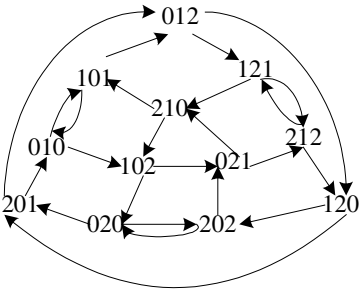


Fig. 1. Kautz graph $K(2,3)$

3 Fission Design

We use Kautz graph $K(2,k)$ as the topology of Fission. Initially, there is a virtual 2-dimensional Cartesian coordinate which is divided into three zones. Like CAN [6], the entire coordinate space is dynamically partitioned among all the peers in the system such that each peer "owns" its specific zone within the whole space. Unlike CAN, each zone in Fission has an identifier and zones are organized as an approximate Kautz graph $K(2,k)$ according to their identifiers. At the beginning the identifiers of the three initial zones are 0, 1 and 2 respectively, as showed in Figure 2.

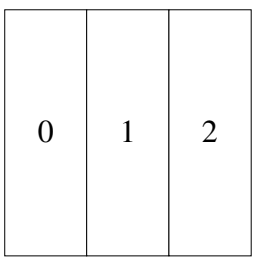


Fig. 2. Initial three Zones

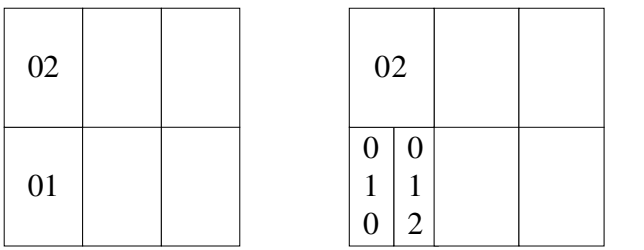
Edges in Fission. The neighborhood of zones is based on their identifiers. The edges between zones can be categorized into three classes: in-edges, out-edges and brother-edges. Suppose the identifier of zone U is $u_1u_2\ldots u_k$ (we denote it as $U=u_1u_2\ldots u_k$), $0\le u_i\le 2$ and $u_i \ne u_{i+1}, 1\le i\le k$. To form a kautz graph, zone U has one out-edge to a zone whose identifier is $u_2u_3\ldots u_i$, $i\le k$ or U has out-edges to zones whose identifier is $u_2u_3\ldots u_kq_{1\ldots}q_m$ with $m\ge 1$. From Lemma 2 proved in section 3.3, we know that the in-edges are only come from zones whose identifiers are $au_1u_2\ldots u_i$ ($a\ne u_1$) with $i\le k$ to

zone U. For the split and mergence of the zones, each zone also has edges to its brothers. Brother-edge will be discussed in section 3.1. At the beginning each zone of the three initial zone has out-edges to the other two zones, and zone 0, 1, 2 are respectively the left brothers of zone 1, 2, 0. We refer to Zone U as the *neighbor* of zone V if there is any edge between U and V.

3.1 Peer Joins

When a new peer n joins in the system, it should know a peer (called *access entry*) currently in the system. As in most DHT systems, we can assume that it can be done with other mechanism, such as using some public available peers.

Spitting Large Zones. When new peer n joins in the system, it firstly chooses a random identifier $U=u_1u_2\ldots u_{128}$ with $u_i \in \{0,1,2\}$, $u_i \neq u_{i+1}$, $1 \leq i \leq 127$. Then peer n sends a JOIN request from the access entry to its destination $u_1u_2\ldots u_{128}$ according to the routing algorithm in section 4. From corollary 4 (also in section 4), the JOIN request will reach a unique zone W whose identifier is the prefix of U at last. Then start from zone W, if the current zone has a neighbor zone with larger area, it forwards the request to the neighbor (If there are more than one neighbors satisfying the condition, select a random one and forward the message to it). This process will not stop until the JOIN request reaches a zone V which has no larger neighbors and can't be forwarded any more. Thus zone V splits itself into two zones V1 and V2 and the original zone V disappears. The owner of zone V1 is set to peer m that was originally the owner of zone V before the split, and the owner of zone V2 is set to peer n. Suppose the identifier of zone V is $v_1v_{2\ldots}v_k$, then the identifier of V1 is $v_1v_{2\ldots}v_kx_0$ and the identifier of V2 is $v_1v_{2\ldots}v_kx_1$ where $0 \leq x_0 < x_1 \leq 2$, $x_0 \neq v_k$ and $x_1 \neq v_k$. After splitting the zone V, both V1 and V2 should build brother-edges to each other: zone V1 is the left brother of V2, and zone V2 is the right brother of zone V1. Figure 3 shows an example of peer n joining into zone 01.



(a)Zones before peer n joins (b)Zones after peer n joins

Fig. 3. Peer n joins in Fission

From the split procedure above, it's easy to know that the area of a zone is in proportion to 2^{-l} when the length of its identifier is l. The longer the identifier is, the less area the zone occupies.

Updating Edges. Once zone V is split, the edges between zones should be updated:

1. in-edge: For each zone U that has an out-edge to zone $V=v_1v_{2...}v_k$ before peer n joins, (From Lemma 3, $U = av_1v_{2...}v_j$ with $j \leq k$) zone U should delete its edge to zone V and adds one out-edge to zone V1 and one out-edge to zone V2.
2. out-edge: For each zone U to which zone V has a out-edge before peer n joins, the area that zone U occupies is no more than that of V because the JOIN message stops at V. Thus $U=v_{2...}v_kq_{1...}q_m$ with $m \geq 1$. Therefore zone V1 adds one out-edge to U if $q_1=x_0$ or V2 adds one out-edge to U if $q_1=x_1$.
3. brother-edge: Suppose the left brother of zone V is U and the right brother of V is Q. After peer n joins, the right brother of U is changed to V1 and the left brother of Q is changed to V2. At the same time, V1 is the left brother of V2 and V2 is the right brother of V1.

After all the related edges have been updated, the Fission network conforms to an approximate Kautz topology again.

3.2 Peer Departs

Merging Small Zones. When a peer n departs from Fission, the zone $V=v_1v_{2...}v_k$ it owns should be occupied by other peers. That will cause the mergence of multiple zones. For good load balance, we try to merge the small zones and maintain the Kautz topology at the same time. If peer n volunteers to depart from the system, it produces a DEPART messages. Start from zone V, if current zone has neighbor zones with smaller area, the DEPART request should be forwarded to one neighbor with smaller zone until a zone U is reached which has no neighbors with smaller area. It is easy to see that U has a brother W whose area is the same as that of U. Suppose $U=u_1u_2\ldots u_{k-1}u_k$, $W=u_1u_2\ldots u_{k-1}w_k$, and the owners of zone U, W are peer n1 and peer n2 respectively. Then:

1. if U=V, merge zone V and W into a new zone $V'= u_1u_2\ldots u_{k-1}$ and assign the peer n2 as the owner of the zone V'. In this case, peer n1 and peer n are the same peer.
2. if U≠V, merge zone U and W into a new zone $V'= u_1u_2\ldots u_{k-1}$, then change the owner of the zone V to peer n1 and assign peer n2 as the owner of zone V'. In this case, the edges related to zone V don't need changing, and only the edges related to zone V' need to be adjusted.

Updating Edges. As discussed above, when zone $U=u_1u_2\ldots u_{k-1}u_k$ and zone $W=u_1u_2\ldots u_{k-1}w_k$ are merged into zone V', the edges related to zone $V'=u_1u_2\ldots u_{k-1}$ should be constructed to retain the approximate Kautz topology. For convenience, suppose $u_k<w_k$, *i.e.*, U is the left brother of W. Then:

1. in-edges: if zone Q has an out-edge to zone U or zone W or both before the mergence, build an edge from zone Q to zone V' to replace it(them).
2. out-edges: if there is an edge from zone U or zone W or both to a zone Q before the mergence, build an edge from V' to Q to replace it (them).
3. brother-edge: if there is a right brother-edge from zone Q to U, build a right brother-edge from Q to V'. If there is a left brother-edge from W to a zone R, build a left brother-edge from V' to R.

Involuntary Departure. To deal with involuntary failure, each peer sends KeepAlive messages to all of its neighbors periodically. The deferred absence of a KeepAlive message from one neighbor indicates its failure. Once the failure of a peer U is detected by its neighbor P, peer P will produce one DEPART message for U. And the remainder procedure is the same as that in the case of voluntary departure.

3.3 Properties of Fission

In this section, we present some properties of neighborhood in Fission. Due to space limits, we just give the brief proofs and omit the complete details.

Lemma 1. *For each zone $U=u_1u_2\ldots u_k$ in Fission, there are no zones V that satisfy $V=u_1u_2\ldots u_kx_1\ldots x_j$ with $j\geq 1$ and $U\neq V$. And each zone in Fission has no less than one out-edge.*

Proof. Initially there are three zones 0, 1, 2 and each zone has two out-edges to the other two zones. So initially lemma 1 holds. After a split or mergence, lemma 1 still holds. Thus lemma 1 is always true. Q.E.D.

Lemma 2. *Denote the length of the identifier of zone U as $|U|$. For each zone $U=u_1u_2\ldots u_k$ in the Fission system, if there is one out-edge from zone U to zone $V=v_1v_{2\ldots}v_m$, then $||U|-|U||\leq 1$, i.e., $|k-m|\leq 1$.*

Proof. Lemma 2 holds initially. We will show that if Lemma 2 holds at some time, Lemma 2 will also hold after a split or mergence. In the case of split, the large zone is divided into two zones. Assume after a split there are two zones U and V with $||U|-|V||\geq 2$ and there is one out-edge from U to V. Recall that before the split for each zone P, W in Fission, $||P|-|W||\leq 1$, thus $||U|-|V||\leq 2$. Therefore $||U|-|V||=2$ and either zone U or zone V is newly produced by the split.

If U is derived from U' in the split, then $||U'|-|V||\leq 1$ and $|U|=|U'|+1$. Obviously to achieve $||U|-|V||=2$, $|U'|-|V|$ must be 1 before the split. Recall that there is one out-edge from U to V after the split, thus U' must have one out-edge to V before the split. But if $|U'|-|V|=1$ (which means the area of zone V is larger than that of zone U') and U' has one out-edge V, then the "split large" policy would cause the JOIN message to be forwarded from U' to V and the zone U' would not have been split. Thus a contradiction occurs.

If V is derived from V' in the split, $|V'|-|U|$ must be 1 before the split. Then zone U is larger than zone V', thus our "split large" policy would not have split the zone V'. Thus a contradiction occurs.

Therefore, after a split Lemma 2 remains true.

In the case of mergence, proof is similar and we omit it for the space.

Q.E.D.

Lemma 3. *For each zone $U=u_1u_2\ldots u_k$, one and only one of the following two properties is true:*

1. *U has a unique out-edge to zone $Q=u_2\ldots u_k$*
2. *For each $x_1\in\{0,1,2\}$ and $x_1\neq u_k$, either U has one out-edge to zone $Q=u_2\ldots u_kx_1$ or U has two out-edges to zones whose identifiers are in the style of $u_2\ldots u_kx_1x_2$ with $x_2\neq x_1$ and $x_2\in\{0,1,2\}$.*

Proof. From lemma 2, if there is one out-edge from Q to U, then $|U|-1 \le |Q| \le |U|+1$. And as the Fission system forms an approximate Kautz topology, thus the identifier of Q should be in the style of $u_2 \ldots u_k$ or $u_2 \ldots u_k x_1$ or $u_2 \ldots u_k x_1 x_2$. The proof is easy but long to make when considering the split and mergence procedure of zones. We omit it for space. Q.E.D.

The following two corollaries are direct conclusions from Lemma 3.

Corollary 1. *If zone $U=u_1u_2\ldots u_k$ has more than one out-edge, then for any string $S=s_1\ldots s_m$, $s_1 \neq u_k$ and $s_i \neq s_{i+1}$ $(1 \le i \le m-1)$, U has an out-edge to zone $u_2\ldots u_k x_1\ldots x_j$ $(1 \le j \le 2)$ with $x_1\ldots x_j$ as the prefix of S.*

Lemma 3 and Corollary1 show that the routing algorithm in section 4 could go on until the destination zone V is reached.

Corollary 2. *The out-degree of any zone in Fission is no more than 4.*

We can also prove that the in-degree of any zone is also no more than 4 in a similar way. The following Theorem 1 is a direct consequence of Lemma 1 and Corollary 2.

Theorem 1. *In an N-peer Fission system, both the in-degree and the out-degree of each peer are between 1 and 4.*

4 Routing Algorithm

Routing in Fission is somewhat similar to that in Kautz graph. Once a zone $U=u_1u_2\ldots u_k$ receives a routing message *Routing*(*V*,*L*,*S*) to destination $V=v_1v_2\ldots v_m$ $(U \neq V)$ with left path length *L*, U sends a new routing message *Routing*(*V*,*L-1*, *S*) to Q if U has one out-edge to zone $Q=u_2u_3\ldots u_k$, and U sends a new routing message *Routing*(*V*,*L-1*,$Sx_1\ldots x_j$) to Q if U has one out-edge to zone $Q=u_2\ldots u_k x_1\ldots x_j$ $(1 \le j \le 2)$ and $Sx_1\ldots x_j$ is the prefix of V. The initial value of L and S is set as below: Assume there is a message routing from source zone $W=w_1w_2\ldots w_k$ to destination zone $V=v_1v_2\ldots v_m$. Find S' that is the longest suffix of W and also appears as a prefix of V and the length of S' is denoted by s. Suppose $W \neq V$, then from Lemma 1, $s<k$. Thus $S'=w_{k-s+1}\ldots w_k$ and the initial value of L is k-s and the initial value of $S=S'=w_{k-s+1}\ldots w_k$. Figure 4 shows the routing algorithm.

```
Procedure U.routing(dest V, pathlegth L, commonfix S)
  //zone U=u1u2…uk deal with the routing
  //message to destination V=v1v2…vm
  {
  /*reach destination V*/
  if L=0 then return(true)
  else {
    if U has a out-edge to Q=u2u3…uk then
       return(Q.routing(V, L-1,S))
```

```
  if U has a out-edge to Q=u2…ukx1…xj (1≤j≤2) and
isprefix(Sx1…xj, V) then {
      S← Sx1…xj
      return(Q.routing(V, L-1,S))
    } //then
  } //else
} //Procedure
```

Fig. 4. Routing algorithm

Now we prove the correctness of the routing algorithm.

Lemma 4. *Considering routing from source zone $W=w_1w_2…w_k$ to any destination $V=v_1v_2…v_m$ ($W\neq V$) where the length of the longest suffix of W that is also the prefix of V is s, let the routing path from W to V is $U_1(=W), U_2, U_3, …, U_q (=V)$, then U_i is of the form $w_i…w_{k-s}S$ and the routing message that U_i deals with is in the form of routing(V,k-s-i,S) where S is the prefix of V.*

Proof. Let S_0 be the longest suffix of W that is the prefix of V, $S_0=w_{k-s+1}…w_k=v_1v_{2…}v_s$, then $U_1=W=w_1w_2…w_{k-s}w_{k-s+1}…w_k=w_1w_2…w_{k-s}S_0$, $V= v_1v_{2…}v_pv_{p+1}…v_m =S_0v_{s+1}…v_m$. The routing message that W deals with is *routing*($V,k-s,S_0$). Thus initially Lemma4 holds for U_1.

Let current zone U_i is of the form $w_i…w_{k-p}S$ and the routing message that U_i deals with is *routing*($V,k-s-i,S$) where S is the prefix of V. From the routing algorithm in Figure 4, if U_i has one edge to $w_{i+1}…w_{k-s}S$, then $U_{i+1}= w_{i+1}…w_{k-s}S$ and the routing message that U_{i+1} deals with is *routing*($V,k-s-i-1,S$); or if U_i has one edge to $w_{i+1}…w_{k-s}Sx_1…x_j$ ($1\leq j\leq 2$) with $S'=Sx_1…x_j$ as the prefix of V, then $U_{i+1}= w_{i+1}…w_{k-s}Sx_1…x_j= w_{i+1}…w_{k-s}S'$ and the routing message that U_{i+1} deals with is *routing*($V,k-s-i,S'$). In both cases Lemma 4 holds for U_{i+1}. So Lemma 4 holds.

Q.E.D.

From Lemma 4, when i=k-s (L=0), the routing messages reach a zone U_i and the routing process stops. And there is some S that is the prefix of V and U_i=S. If destination V is a zone identifier, then from Lemma 1 we can infer that U_i=S=V and the routing reaches the destination zone correctly. The path length of routing from W to V is k-s hops. Thus we get the following corollaries.

Corollary 3. *The path length of routing initiated by any source zone $U=u_1u_2…u_k$ is no more than k hops.*

From the proof of Lemma 4, we can get the following corollary 4 easily.

Corollary 4. *For any identifier $U=u_1u_2…u_{128}$ with $u_i\in\{0,1,2\}$, $u_i\neq u_{i+1}$, $1\leq i\leq 127$, the routing from any source zone to destination U will arrive and stop at a unique zone V whose identifier is the prefix of U.*

Lemma 5. *In an N-peer Fission system, the largest zone U satisfies that $|U|<logN$.*

Proof. Let |U|=k, then k is the smallest among the length of identifiers of zones in Fission. Peers in Fission form an approximate Kautz topology, thus $2^k + 2^{k-1} \leq N$. Then $k \leq \log N - \log 3 + 1 < \log N$. Q.E.D.

Lemma 6. *In an N-peer Fission system, the smallest zone V satisfies* $|V| < 2*\log N$.

Proof. Suppose U is the largest zone in Fission system. Consider the routing path from U to V. From Corollary 3, we know that the path length is no more than |U|. From Lemma2, we can infer that $||V|-|U|| \leq |U|$. Because $|V| \geq |U|$, thus $|V|-|U| \leq |U|$, $|V| \leq 2|U| < 2\log N$. Q.E.D.

The following can be derived from Lemma 6 and Corollary 3 directly.

Corollary 5. *In an N-peer Fission system, Let U and V be the smallest and largest zones in the system, then* $|U|-|V| \leq |V| < \log N$.

The following Theorem 2 is a direct consequence of Corollary 3 and Lemma 6.

Theorem 2. *The diameter of Fission systems is less than 2*log N.*

Theorem 3. *When a peer joins in or departs from an N-peer Fission system, the JOIN and DEPART messages are propagated at most 3*logN and logN hops respectively, and only constant peers need to update their edges.*

Proof. Take the split procedure as an example: the JOIN message is first forwarded to the zone determined by hashing. From Theorem 2, in this phase the message is propagated less than $2*\log N$ hops. Then the JOIN message is forwarded to neighbors whose identifiers are at least one less than that of current zone. From Corollary 5, in the phase the message is propagated at most $\log N$ hops. Therefore, the JOIN message is propagated at most $3*\log N$ hops. When the JOIN message stops at one zone, from Theorem 1 the number of neighbors that should update edges due to the split is constant. Q.E.D.

5 Performance Evaluations

Firstly we analyze the performance of Fission by the measures below:

Degree: From Theorem 1, Fission system is of constant-degree, i.e., is $O(1)$.

Diameter: From Theorem 2, the diameter of Fission is less than $2*\log N$, i.e., $O(\log N)$.

Scalability: Fission can support arbitrary number of peers. Peer can join in the system and depart from the system at its will.

Maintenance cost: From Theorem 3, the events of a peer joining in or departing from the Fission system induce $O(\log N)$ messages and require $O(1)$ peers to change their state.

We also built a simulator to evaluate the Fission algorithm. The simulator implements the joining and departing procedure in Section 3 and uses the routing algorithm in Section 4. We evaluate the average path length of Fission systems of different scales (from 256 peers up to 256K peers). For each scale, we build the network by adding nodes one by one. We select two random points from the 2-

dimention coordinate space and route a message from one point to the other, and get the average path length over 5000 such routings. Figure 5 shows the simulation results compared with CAN with d=2 or d=3. From Figure 5, we can infer that the average path length of Fission is about log*N*.

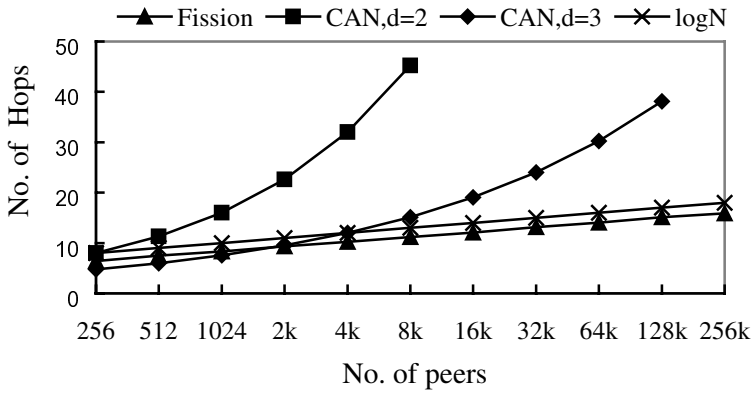


Fig. 5. Average path length of Fission

Then we observe the load balance characteristic of Fission. Let the total area of the entire 2-dimensition coordinate space is S_T. We simulate Fission system with N=4096 peers and calculate the area each peer owns. Let $S=S_T/N$, Figure 6 shows the percentage of peers that own a particular area. From Figure 6, about 45% peers own the same area S and the percentage that owns area more than 4S or less than S/4 is almost zero. The number of keys stored on each peer is in direct proportion to the area the peer owns, thus the distribution of keys over peers is almost uniform.

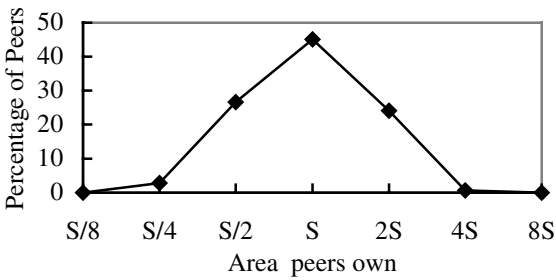


Fig. 6. Load balance characteristic of Fission

6 Conclusions and Future Work

Fission is a novel scalable DHT-style peer-to-peer network that can achieve O(log$N$) diameter with only O(1) degrees. Beside its simplicity and low diameter as well as constant degree, Fission also has good load balance characteristic. These

characteristics of Fission suggest that Fission is a very promising peer-to-peer network. In our further work, physical topological information will be exploited in Fission to reduce latency. And the current design of Fission is based on Kautz graph $K(2,k)$ and will be extended to general $K(d,k)$ topology for flexibility.

References

1. Clark, D.: Face-to-face with peer-to-peer networking. IEEE Computer, Vol. 34, No.1, IEEE press (2001) 18–21
2. Schoder, D., Fischbach, K.: peer-to-peer prospects. Communications of the ACM, Vol.46, No.2, (2003) 27–29
3. Stoica, I., Morris, R., Karger. D. *et al.*: Chord: a scalable peer-to-peer lookup service for Internet applications. Proc. of ACM SIGCOMM 2001, ACM Press, New York (2001) 160–177
4. Hildrum, K., Kubiatowicz, J. D., Rao, S., and Zhao, B. Y.: Distributed object location in a dynamic network. Proc. of 14th ACM Symp. on Parallel Algorithms and Architectures (SPAA) (2002)
5. Rowstron, A. and Druschel, P.: Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. Proc. of IFIP/ACM Middleware'2001, Heidelberg, Germany (2001) 329–350.
6. Ratnasamy, S., Francis, P., Handley, M. *et al.*: A scalable content-addressable network. Proc. of ACM SIGCOMM 2001, ACM Press, New York (2001) 149–160
7. Plaxton, C., Rajaraman, R., and Richa, A.: Accessing nearby copies of replicated objects in a distributed environment. Proc. of ACM Symp. on Parallel Algorithms and Architectures (SPAA), Newport, Rhode Island (1997).
8. Balakrishnan, H., Kaashoek, M. F., Karger, D., Morris, R., and Stoica, I.: Looking up data in P2P systems. Communication of the ACM, Vol. 46, No.2, (2003) 43–48
9. Fraigniaud, P., Gauron, P.: The Content-Addressable Network D2B. Tech Rept. 1349, CNRS University paris-Sud, France (2003)
10. Malkhi, D., Naor, M., and Ratajczak, D.: Viceroy: a scalable and dynamic lookup network. Proc. of 21st AC M Symp. on Principles of Distributed Computing (PODC), Monterey, CA (2002)
11. Kautz, W. H.: The design of optimum interconnection networks for multiprocessors. Architecture and design of Digital computer. NATO advances summer Institute, (1969) 249–277
12. Panchapakesan, G. and Sengupta, A.: On a lightwave Network Topology using Kautz Digraphs. IEEE Transaction on computers, Vol. 48, No. 10, (1999) 1131–1138
13. Sivarajan, K. N. and Ramaswami, R.: Lightwave Networks based on de Bruijn Graphs. IEEE/ACM Trans. Networking, Vol.2 (1994) 70–79