Distributed Line Graphs: A Universal Framework for Building DHTs Based on Arbitrary Constant-Degree Graphs

Yiming Zhang[1], Ling Liu[2], Dongsheng Li[1], Xicheng Lu[1]
[1]*National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha, China*
[1]*{ymzhang, dsli, xclu}@nudt.edu.cn*
[2]*College of Computing, Georgia Institute of Technology, Atlanta, USA*
[2]*lingliu@cc.gatech.edu*

Abstract

Most proposed DHTs have their unique maintenance mechanisms specific to the static graphs on which they are based. In this paper we propose distributed line graphs (DLG), a universal framework for building DHTs based on arbitrary constant-degree graphs. We prove that in a DLG-enabled, N-node DHT, the out-degree is d, the in-degree is between 1 *and* 2*d, and the diameter is less than* $2(\log_d N - \log_d N_0 + D_0 + 1)$, *where d, D_0 and N_0 represent the degree, diameter and number of nodes of the initial graph, respectively. The maintenance cost of DLG-enabled DHTs is* $O(\log_d N)$. *We show the power of DLG technique by applying it to Kautz graphs to propose a new DHT scheme.*

1. Introduction

Recently constant-degree Distributed Hash Tables (DHTs) have attracted significant attention from both industry and academic research due to their good tradeoffs between routing path length and routing table size. Constant-degree DHTs are usually proposed based on a specific type of constant-degree graphs. E.g., CAN [1] is based on d-torus; Viceroy [2] is based on butterfly; D2B [3] and Koorde [4] are based on de Bruijn graphs; FissionE [5] and Moore [6] are based on Kautz graphs.

DHTs engage certain *maintenance mechanisms* to deal with dynamics such as node join/departure. These mechanisms are usually complicated and error-prone. Most constant-degree DHTs have a clean-slate design of their maintenance mechanisms, tightly coupled with the static graphs on which they are based. In this paper, we present the *distributed line graphs* (*DLG*) technique, a universal framework for building DHTs based on arbitrary constant-degree graphs.

We prove that in a DLG-enabled, *N*-node DHT, the out-degree is *d*, the in-degree is between 1 and 2*d*, and the diameter is less than $2(\log_d N - \log_d N_0 + D_0 + 1)$, where *d*, D_0 and N_0 represent the degree, diameter and number of nodes of the initial graph, respectively. This diameter reaches the theoretical lower bound $\Omega(\log_d N)$ of constant-degree DHTs [7]. The number of messages caused by each node join/leave is $O(\log_d N)$.

Our proposed DLG technique is inspired by a novel technique called *line graph* (*LG*) iteration [8]. The LG iteration, as well as its variations [8-11], has been extensively studied and used as a universal framework for building different multiprocessor networks. However, it requires global topology information and centralized control, and thus cannot be applied to DHTs.

The challenge we face is the absence of knowledge for what graphs we are to deal with, combined with the characteristics of DHTs such as decentralized control and lack of global information. To address this, the key idea behind our DLG-based solution lies in its iterated edge-node transition for routing table construction, which retains most properties of the initial graphs.

To the best of our knowledge, we are the first to propose a universal framework for building DHTs based on arbitrary constant-degree graphs. Using the DLG technique, only minimum extra work is required to do to design constant-degree DHTs. We demonstrate the power of DLG technique by applying it to Kautz graphs to present *DLG-Kautz* (*DK*), a DLG-enabled and Kautz graph-based DHT. Furthermore, we use DK as a model to evaluate the effectiveness of our proposals through extensive simulations, and compare it with other DHTs.

2. Concept and line graph iteration

A *graph* $G = (V, E)$ consists of a set of *nodes* $V=V(G)$ and a set of directed *edges* $E=E(G)$ between nodes.

If $a = [x, y]$ is an edge from x to y, we say that x (or a) is *adjacent to* y and y is an *out-neighbor* of x, and also that y (or a) is *adjacent from* x and x is an *in-neighbor* of

1063-6927/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDCS.2008.35

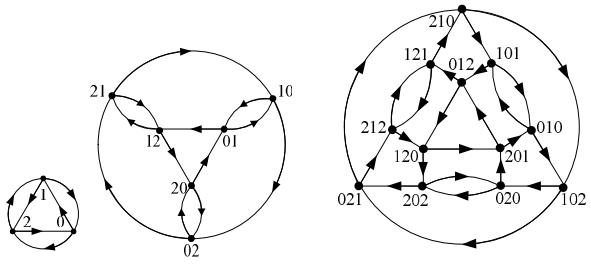
y. The number of nodes in G is called the *order* of G.

Let $\Gamma_G^-(x)$ and $\Gamma_G^+(x)$ denote the sets of nodes adjacent to and from x, respectively. Their cardinalities are the *in-degree*, $\delta_G^-(x)=|\Gamma_G^-(x)|$, and the *out-degree*, $\delta_G^+(x)=|\Gamma_G^+(x)|$, of x. Graph G is *d-out-regular* (resp. *d-in-regular*) if $\delta_G^+(x)=d$ (resp. $\delta_G^-(x)=d$) for all nodes $x\in G$. Graph G is *d-regular* (*constant-degree*) if it is *d-out-regular* and *d-in-regular*.

The *diameter* of G is the largest distance over all the pairs of nodes. A series of graphs is *incrementally expandable* if it contains graphs of size n for any $n\geq n_0$.

Let the initial graph G_0 be a d-regular graph. In the *line graph* [8] $G_{i+1}=L(G_i)$ of G_i, $i=0,1,2,\ldots$, each node corresponds to an edge of G_i, i.e. $V(G_{i+1})=\{uv|[u,v]\in E(G_i)\}$; and a node uv is adjacent to a node wz iff $v=w$, that is, $[uv,wz]\in G_{i+1}$ when edge $[u,v]$ is adjacent to edge $[w,z]$ in G_i. Clearly G_i, $i=0,1,2,\ldots$, is d-regular.

Line graphs can be defined iteratively as $G_i=L(G_{i-1})=\ldots=L^i(G_0)$. *Line graph* (*LG*) *iteration* simply refers to the process of generating graphs by iteratively applying the L operator, and G_i is said to be *derived from* G_0. Figure 1 shows two examples of LG iterations.



a) $G_0=K(2,1)$ *b*) $G_1=L(G_0)=K(2,2)$ *c*) $G_2=L(G_1)=K(2,3)$

Figure 1. **Examples of line graph iterations**

Let the order and diameter of G_0 be N_0 and D_0. By [8], the order N_n and diameter D_n of $G_n=L^n(G_0)$ satisfy

$$N_n=N_0\times d^n,\quad D_n=D_0+n \tag{1}$$

LG iteration retains almost all properties of the initial graph, such as the degree, diameter, connectivity, and routing algorithm. Many graphs can be defined iteratively by LG iterations. For example, Kautz graph $K(d,D)=L(K(d,D-1))=\ldots=L^{D-1}(K(d,1))$. However, it is easy to see they are not *incrementally expandable*.

To support any number of nodes, many variations were proposed, e.g. partial line graphs (PLG) [8], necklaces [9], and factorization [10]. However, they are centralized algorithms and require global topology information, and thus cannot be applied to DHTs.

3. Distributed line graphs

3.1 Basic DL graphs and DL iteration

Different graphs usually utilize different notation to define their nodes and edges. We first unify the way to describe a d-regular, N_0-node initial graph G_0.

Let $|u|$ denote the length of identifier of node u, e.g. if $u=u_1u_2\ldots u_k$, then $|u|=k$. Let X be an *alphabet* of N_0 letters. Clearly $\forall x\in X$ satisfies $|x|=1$. The initial graph G_0 has its nodes labeled with letters $x\in X$ one by one, and for each node $\alpha\in G_0$ there is an *in-letter set* $\zeta(\alpha)$ and an *out-letter set* $\psi(\alpha)$, which are defined as

$$\zeta(\alpha)=\Gamma_{G_0}^-(\alpha),\ \psi(\alpha)=\Gamma_{G_0}^+(\alpha). \tag{2}$$

Obviously $|\zeta(\alpha)|=|\psi(\alpha)|=d$. Since all initial graphs illustrated in this paper have an order of less than 10, we simply label the nodes in an initial graph G_0 with identifiers $0,1,\ldots,N_0-1$. This alphabet approach enables us to describe all d-regular graphs in a universal way. For example, Figure 2 shows a (2,2)-butterfly graph, in which the in-letter set and out-letter set of node 0 are $\zeta(0)=\{4,6\}$ and $\psi(0)=\{4,5\}$, respectively.

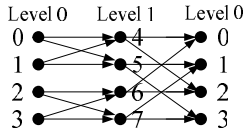


Figure 2. **Universal naming of a butterfly graph**

Let $u=u_1u_2\ldots u_m$, $v=v_1v_2\ldots v_n$, $m\geq n$. Define

$$u\circ v=u_{m-n+1}v. \tag{3}$$

The technique presented in this paper is based on the concept of *distributed line graphs* as follows.

Definition 1: Let the initial graph $G_0=(V,E)$ be a d-regular graph. A series of graphs $G_{i+1}=DL(G_i,v^{(i)})$, where $v^{(i)}\in V(G_i)$ satisfies

$$\forall u\in\Gamma_{G_i}^-(v^{(i)})\cup\Gamma_{G_i}^+(v^{(i)})\quad |v^{(i)}|\leq|u| \tag{4a}$$

for $i=0,1,2,\ldots$, are said to be a family of *distributed line* (*DL*) *graphs* with base d derived from the *initial graph* G_0, if the following conditions hold.

$$V(G_{i+1})=V(G_i)-\{v^{(i)}\}+\{u\circ v^{(i)}\,|\,u\in\Gamma_{G_i}^-(v^{(i)})\} \tag{4b}$$

$$\begin{aligned}E(G_{i+1})=&E(G_i)-\{[x,v^{(i)}]\,|\,x\in\Gamma_{G_i}^-(v^{(i)})\}-\{[v^{(i)},y]\,|\,y\in\Gamma_{G_i}^+(v^{(i)})\}+\\&\{[u,u\circ v^{(i)}]\,|\,u\in\Gamma_{G_i}^-(v^{(i)})\}+\{[u\circ v^{(i)},w]\,|\,u\in\Gamma_{G_i}^-(v^{(i)}),w\in\Gamma_{G_i}^+(v^{(i)})\}\end{aligned} \tag{4c}$$

The transitions from G_i to G_{i+1} are called *distributed line* (*DL*) *iteration*. Node $v^{(i)}$ is called the *responsible node* for $G_{i+1}=DL(G_i,v^{(i)})$. Note the responsible nodes of DL iterations are constrained to the ones with locally shortest identifier compared with their neighbors. A new node $u\circ v^{(i)}\in G_{i+1}$ corresponds to an edge $[u,v^{(i)}]\in G_i$.

The DL iteration from G_i to G_{i+1} can be summarized as follows. 1) Let G_{i+1} and G_i be the same, i.e. $G_{i+1}=G_i$; 2) delete node $v^{(i)}$ and all edges adjacent to/from $v^{(i)}$ in the new graph G_{i+1}; 3) for each edge $[u,v^{(i)}]$ adjacent to

$v^{(i)}$ in G_i, add a new node $u \circ v^{(i)}$ to G_{i+1}; and 4) in G_{i+1}, the edges of the form $[u, u \circ v^{(i)}]$ will be adjacent to $u \circ v^{(i)}$, and the edges of the form $[u \circ v^{(i)}, w]$ will be adjacent from $u \circ v^{(i)}$ with $w \in \Gamma^{+}_{G_i}(v^{(i)})$.

Figure 3 shows three examples of DL iterations.

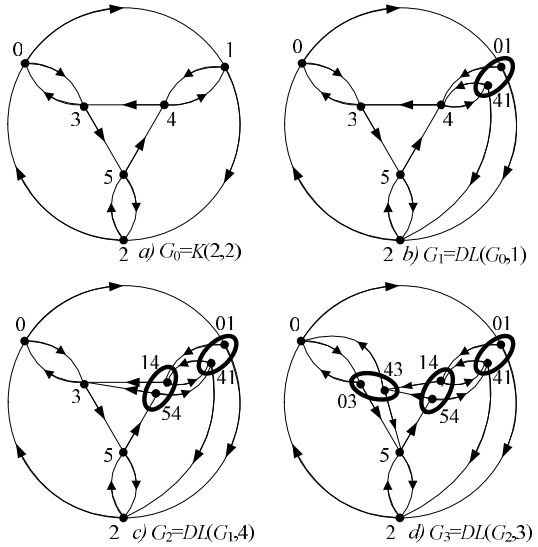


Figure 3. **Examples of DL iterations**

To analyze the in-neighbors and out-neighbors of a node (in Theorems 1 and 2), next we present Lemmas 1~2. Note that in this paper we only present the proofs of important theorems due to lack of space.

Lemma 1: Let graph G be a DL graph with base d. Let $x = x_1x_2\ldots x_n \in V(G)$. Then there are no nodes of the form $x' = x_ix_{i+1}\ldots x_n$ in graph G, where $1< i \le n$.

Lemma 1 shows that in a DL graph the identifier of one node can not be the suffix of another.

Lemma 2: Let graph G be a DL graph and $x \in V(G)$. If there is some node $y \in \Gamma^{-}_{G}(x)$ satisfying $|y|<|x|$, then

$$\delta^{-}_{G}(x) = 1. \tag{5}$$

Lemma 2 shows that if a node x has an in-neighbor with shorter identifier compared with x, then it has no other in-neighbors. Clearly, by Lemma 2 it is easy to see that if there is some node $y \in \Gamma^{-}_{G}(x)$ satisfying $|y|\ge|x|$, then for any node $y \in \Gamma^{-}_{G}(x)$ we have $|y|\ge|x|$.

The following Theorem 1 describes the in-neighbors of a node in DL graphs.

Theorem 1: Let graph G be a DL graph with base d. Let $x = x_1x_2\ldots x_n \in V(G)$.

If there is a node $y \in \Gamma^{-}_{G}(x)$ satisfying $|y|<|x|$, then

$$\Gamma^{-}_{G}(x) = \{y \mid y = x_1 \ldots x_{n-1} \in V(G)\}. \tag{6a}$$

Otherwise for each $\beta \in \zeta(x_1)$ (defined in (2)), either there is one in-neighbor $y \in \Gamma^{-}_{G}(x)$ satisfying

$$y = \beta x_1 x_2 \ldots x_{n-1}, \tag{6b}$$

or there are d in-neighbors $y \in \Gamma^{-}_{G}(x)$ satisfying

$$y = \beta' \beta x_1 x_2 \ldots x_{n-1} \tag{6c}$$

with $\beta' \in \zeta(\beta)$. And x has no other in-neighbors in G.

Proof: Theorem 1 holds initially for G_0. Suppose Theorem 1 holds for G_i. Let $v=v_1v_2\ldots v_m$, $G_{i+1}=DL(G_i, v)$.

We first prove Theorem 1 holds if x is a new node. By (3), the new nodes are of the form $\alpha v_1v_2\ldots v_m$.

If there is some node $y \in \Gamma^{-}_{G_{i+1}}(x)$ satisfying $|y|<|x|$, then by (4a) we have $|y|=|u|$. By (6b) we have

$$y = \alpha v_1 v_2 \ldots v_{m-1}. \tag{7}$$

By Lemma 2, node y is the only in-neighbor of x and thus (6a) holds in this case.

Otherwise, similarly to the above case, by (3), (4a) and (6a), the in-neighbors of v in G_i are of only one form: $u'=\alpha'\alpha v_1v_2\ldots v_{m-1}$. Clearly we have $|u'| = |x| = |v|+1$. Since Theorem 1 holds for G_i, we have $\alpha = u'_2 \in \zeta(v_1)$ in G_i, and node v has d distinct in-neighbors in the form of $y = \lambda\alpha v_1 v_2 \ldots v_{m-1}$ with $\lambda \in \zeta(\alpha)$. According to (4c), they are the in-neighbor set of x in G_{i+1}. So, Theorem 1 holds when x is a new node.

Similarly, Theorem 1 still holds for graph G_{i+1} when x is an out-neighbor of node v.

Compare G_{i+1} with G_i, the in-neighbors of other nodes do not change. Therefore, Theorem 1 holds. ■

From Theorem 1 it is easy to infer Corollaries 1~3.

Corollary 1: If two nodes x and y are neighbors in a DL graph, then $||x| - |y|| \le 1$.

Corollary 1 describes the relationship between the identifier lengths of any two neighbors in a DL graph.

Corollary 2: Let graph G be a DL graph with base d. If $G' = DL(G, v)$, then we have

$$|V(G')| - |V(G)| = d - 1, \tag{8a}$$

$$V(G') - V(G) = \{\alpha v_1 v_2 \ldots v_n\} \text{ with } \alpha \in \zeta(v_1). \tag{8b}$$

Corollary 2 shows that there would be $d-1$ new nodes generated after the DL iteration.

Corollary 3: Let graph G be a DL graph with base d. Let $x = x_1x_2\ldots x_m \in V(G)$. Then for $i = 1,2,\ldots, m-1$,

$$x_i \in \zeta(x_{i+1}),\ x_{i+1} \in \psi(x_i) \tag{9}$$

Corollary 3 describes the relationship between any two consecutive letters in a node identifier in DL graphs.

The following Theorem 2 describes the out-neighbors of a node in DL graphs.

Theorem 2: Let graph G be a DL graph with base d. Let $x = x_1x_2\ldots x_n \in V(G)$. For each $\beta \in \psi(x_n)$, there is one out-neighbor $y \in \Gamma^{+}_{G}(x)$ satisfying

$$y = x_t \ldots x_2 x_3 \ldots x_n \beta \tag{10}$$

with $1 \le t \le 3$. And x has no other out-neighbors in G.

Note that $x_m\ldots x_n$ represents a null string if $m>n$. The proof of Theorem 2 is similar to that of Theorem 1.

The following Theorem 3 describes the in-degree and out-degree of a node in DL graphs.

Theorem 3: Let graph G be a DL graph with base d. For each node $x \in V(G)$, the out-degree and in-degree, $\delta_G^+(x)$ and $\delta_G^-(x)$, satisfy respectively

$$\delta_G^+(x) = d\ ,\ 1 \le \delta_G^-(x) \le d^2\ . \tag{11}$$

And the average degree of all nodes is $2d$.

Proof: $\delta_G^+(x) = d$ holds initially for G_0. Suppose it holds for G_i with $i \ge 0$. From Definition 1 the new nodes and old nodes in $G_{i+1} = DL(G_i, v)$ all have an out-degree of d. Thus $\delta_G^+(x) = d$ holds.

If node x has one in-neighbor y satisfying $|y|<|x|$, then by Lemma 2 we have $\delta_G^-(x) = 1$. Otherwise each in-neighbor y satisfies $|y| \ge |x|$. Then by Theorem 1 we have $d \le \delta_G^-(x) \le d^2$. Thus $1 \le \delta_G^-(x) \le d^2$ holds.

Obviously, the in-degrees and out-degrees of all nodes in graph G satisfy

$$\sum_{x \in V(G)} \delta_G^+(x) = \sum_{x \in V(G)} \delta_G^-(x) = |E(G)|\ . \tag{12}$$

So the average degree is $2d$, and Theorem 3 holds. ■

To get the upper bound for the diameter of a DL graph, we first define the *optimal* DL iteration and φ *mapping*, and present the following Lemmas 3, 4 and 5.

Let the initial graph $G_0 = (V, E)$ be a constant-degree graph. For i = 0,1,2,…, DL iterations $G_{i+1} = DL(G_i, v^{(i)})$ are said to be *optimal*, if condition (4a) for responsible node $v^{(i)}$ in Definition 1 is tightened as

$$\forall u \in V(G_i)\quad |v^{(i)}| \le |u|\ . \tag{13}$$

Note that by (13) the identifier of the responsible node is required to be globally shortest, while by (4a) it is only required to be locally shortest compared with its direct neighbors. Let u, v be any two nodes in graph G derived by a series of *optimal* DL iterations. By (13), obviously we have $||u| - |v|| \le 1$.

Let $u = u_1u_2\ldots u_m$. Define

$$\varphi(u,i) = \begin{cases} u_{m-i+1}..u_m, & \text{if } i \le m \\ u, & \text{otherwise.} \end{cases} \tag{14}$$

Clearly, if $i \le m$ then $|\varphi(u,i)| = i$, and if $|u| \ge |v|$ then $\varphi(u \circ v, |v|)=v$. E.g. $\varphi(012,2) = 12$, $\varphi(012 \circ 23, |23|) = 23$.

Define φ *mapping* of G as $G' = \varphi(G,m)$, where

$$V(G') = \{\varphi(v,m)\,|\,v \in V(G)\}\ , \tag{15a}$$

$$E(G') = \{[\varphi(v,m),\varphi(w,m)]\,|\,[v,w] \in E(G)\} \tag{15b}$$

Clearly the orders of G and G' satisfy $|G| \ge |G'|$. For example, in Figure 3 we have $G_0=\varphi(G_i,1)$ with i=1,2,3.

Lemma 3: Let graph G be a DL graph derived from G_0 with base d and diameter D_0. Let u and v be any two nodes in G. The distance from u to v, $d_G(u, v)$, satisfies

$$d_G(u,v) \le |v| + D_0 - 1\ . \tag{16}$$

Lemma 3 describes the distance between any two nodes in a DL graph. For example, in a DL graph initiated from Kautz graph $K(d,1)$, the path length is no greater than the identifier length of the destination node.

Lemma 4: Let graph G be a DL graph derived from the initial graph G_0 by a series of *optimal* DL iterations. Let the shortest node in G be of length m. Then

$$\varphi(G,m) = L^{m-1}(G_0)\ . \tag{17}$$

Lemma 4 shows that a DL graph derived by *optimal* DL iterations is isomorphic to a standard line graph under the φ *mapping*. The correctness of Lemma 4 lies in the *homomorphism* of the *optimal DL* iterations and the *PLG* iterations [8].

Lemma 5: Let graph G be a DL graph derived from the initial graph G_0 by a series of DL iterations. Let the shortest node identifier in G be of length n. Then

$$\varphi(G,n) = L^{n-1}(G_0)\ . \tag{18}$$

Lemma 5 shows that a DL graph derived by normal DL iterations is also isomorphic to a standard line graph under the φ *mapping*. The following Theorem 4 gives the upper bound for the diameter of a DL graph.

Theorem 4: Let graph G be a DL graph with base d and order N. Let the order and diameter of the initial graph G_0 be N_0 and D_0, respectively. Then the diameter of G, $D(G)$, satisfies

$$D(G) \le 2(\log_d N - \log_d N_0 + D_0)\ . \tag{19}$$

Proof: Let u and v be of the shortest and longest identifier in G, respectively. Let $|u| = m$, $|v| = n$.

From Lemma 3, we have

$$D(G) \le D_0 + n - 1\ . \tag{20}$$

By Lemma 5 we have $\varphi(G,m) = L^{m-1}(G_0)$. Let $N(G)$ denote the order of graph G. By (1), (15a) we have

$$N(G) \ge N(L^{m-1}(G_0)) = N_0 \times d^{m-1}\ . \tag{21}$$

The distance from v to u in G, $d_G(v, u)$, satisfies

$$d_G(v,u) \le D_0 + m - 1\ . \tag{22}$$

From Corollary 1, if two nodes x and y are neighbors in G, then $||x| - |y|| \le 1$. Let the path from v to u in G is

$$v \to x^{(1)} \to x^{(2)} \to \ldots \to x^{(j)} \to u\ . \tag{23}$$

Then the identifier lengths of node v and u satisfies

$$\begin{aligned} n - m &= |v| - |u| = (|v| - |x^{(1)}|) + (|x^{(1)}| - |x^{(2)}|) \\ &+ \ldots + (|x^{(j)}| - |u|) \le 1 + 1 + \ldots + 1 = d_G(v,u) \end{aligned} \tag{24}$$

Then, by (20) ~ (22) and (24), Theorem 4 holds. ■

3.2 DL+ graphs with node merge/split

From Corollary 2, the order of $G_{i+1}=DL(G_i, v)$ is $d-1$ greater than that of G_i, which means DL graphs with $d > 2$ are not *incrementally expandable*. E.g., suppose G_0 is a 3-regular graph $G_0=K(3,2)$, as shown in Figure 4(a). After iteration $G_1=DL(G_0,1)$, there would be two more nodes in G_1 than in G_0, as shown in Figure 4(b).

To address this problem, we present the *merge*/*split* operation. Let graph G be a DL graph with base d. From Corollary 2, if $G'=DL(G, v)$, $v=v_1v_2\ldots v_m$, then there would be d distinct new nodes in G' and $V(G') - V(G) = \{\alpha v_1v_2\ldots v_m\}$ with $\alpha \in \zeta(v_1)$. Sort them by ascending their first letters, then the d new nodes are of the form

$\alpha_i v_1 v_2 \ldots v_m$ with $\alpha_i \in \zeta(v_1)$, $i = 0,1,\ldots, d-1$. Each of the d nodes is called *sib-neighbor* to other $d-1$ nodes.

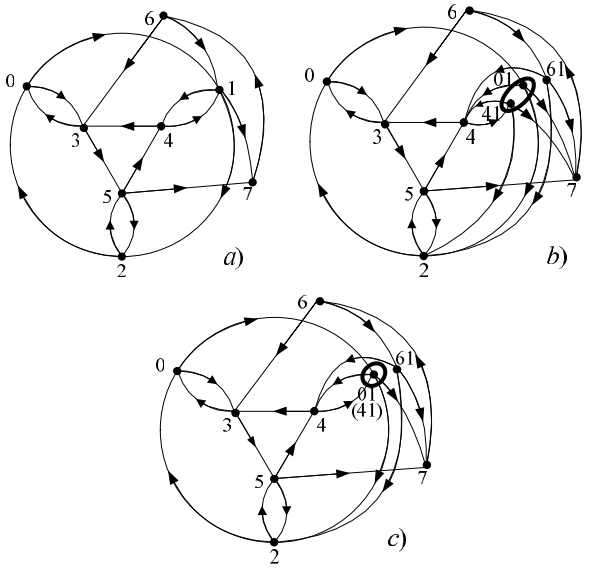


Figure 4. **Example of merge operations**

Node merge. Of d new nodes in $G' = DL(G, v)$, the first half, nodes of the form $\alpha_i v_1 v_2 \ldots v_m$ with $i \le \lceil d/2 \rceil$, merge to one node $s = \beta v_1 v_2 \ldots v_m$ with $\beta = \alpha_0$; and the second half, nodes of the form $\alpha_i v_1 v_2 \ldots v_m$ with $i > \lceil d/2 \rceil$, merge to one node $t = \beta v_1 v_2 \ldots v_m$ with $\beta = \alpha_{\lceil d/2 \rceil + 1}$. This operation is referred to as *merge operation* $G'' = Merge(G', v)$. Node s in G'' holds its component nodes by taking charge of their edges in G', i.e. $\Gamma^-_{G''}(s) = \cup \Gamma^-_{G'}(\alpha_i v_1 v_2 \ldots v_m)$ and $\Gamma^+_{G''}(s) = \cup \Gamma^+_{G'}(\alpha_i v_1 v_2 \ldots v_m)$ with $i \le \lceil d/2 \rceil$, and so does node t.

In this paper, nodes s and t are referred to as *physical nodes*, corresponding to real nodes in a DHT overlay; and nodes that s and t hold are referred to as *logical nodes* (nodes for short), corresponding to nodes in a *DL graph*. Clearly a physical node and its logical nodes have the same identifier length. In the following, the number of logical nodes that a physical node v holds is denoted as $[\![v]\!]$. Figure 4(c) shows an example of the *merge* operation (01 and 41 merge to 01) with $[\![01]\!] = 2$.

Node split. On the other hand, if before the DL iteration $G' = DL(G, v)$, node v satisfies $[\![v]\!] > 1$, then a *split operation* $G' = Split(G, v)$ will take place instead of the DL iteration. Let node $v = \alpha_i v_1 v_2 \ldots v_m$, then the *split* operation divides the logical nodes and their edges that v holds into two shares. The original physical node v holds the first share, the nodes of the form $\alpha_i v_1 v_2 \ldots v_m$ with $j \le i \le j + \lceil [\![v]\!]/2 \rceil$; and a new physical node holds the second share, the nodes of the form $\alpha_i v_1 v_2 \ldots v_m$ with $j + \lceil [\![v]\!]/2 \rceil < i < j + [\![v]\!]$. The new physical node is of the form $\beta v_1 v_2 \ldots v_m$ with $\beta = \alpha_{j + \lceil [\![v]\!]/2 \rceil + 1}$.

We refer to the basic DL iteration combined with the *merge* and *split* operation as *DL+ operation*, which is summarized in Figure 5.

Procedure DL+ operation (OldGraph ***G***, ResultGraph **G'**)
1 **Choose** a physical node $v \in V(G)$ **satisfying** $|v| < |u|$ **or** $(|v| = |u|) \wedge ([\![v]\!] > [\![u]\!])$ for any $u \in \Gamma^-_G(v) \cup \Gamma^+_G(v)$.
2 **if** ($[\![v]\!] > 1$) { $G' = Split(G, v)$; }
3 **else** { $G' = DL(G, v)$; $G' = Merge(G', v)$; }

Figure 5. **DL+ operations**

The graphs generated by the DL+ operations are referred to as *DL+ graphs*. Clearly, the nodes in DL+ graphs are physical nodes, and the nodes in DL graphs are logical nodes. By replacing all physical nodes with their logical nodes, each DL+ graph has a *corresponding* DL graph. If every physical node in a DL+ graph holds only one logical node, the DL+ graph is *isomorphic* to its corresponding DL graph. We say the initial graph, its DL graphs and DL+ graphs have the same base d.

Obviously, the DL+ operation requires only direct neighbor information of node v, and the family of DL+ graphs with arbitrary base is *incrementally extendable*.

The following Theorem 5 describes the in-degree and out-degree of a node in DL+ graphs.

Theorem 5: Let graph G be a DL+ graph with base d. Let the order and diameter of the initial graph G_0 be N_0 and D_0 respectively. For each physical node $x \in V(G)$,

$$\delta^+_G(x) = d\ ,\ 1 \le \delta^-_G(x) \le 2d\ . \qquad (25)$$

And the average in-degree of all nodes is d.

Proof: A physical node x in G is responsible for some logical nodes, which obviously have the same out-neighbors in the corresponding DL graph. Then x has the same out-degree as any of its logical nodes. From Theorem 3 we have $\delta^+_G(x) = d$.

For $\delta^-_G(x)$, consider the following two cases.

1) There is some physical node $v \in \Gamma^-_G(x) \cup \Gamma^+_G(x)$ satisfying $|v| > |x|$. Clearly there must be some certain time when node v is going to be generated by operation $G_{i+1} = DL(G_i, v')$. Note x is a neighbor of v'. Then

$$[\![x]\!] \le [\![v']\!] = 1\ . \qquad (26)$$

Since in G we have $|x| < |v|$, by now (26) still holds. So we would use the notation "x" to represent the logical node of x. Let G^* be the corresponding DL graph of G. From Theorem 1, given any $\alpha \in \zeta(x_1)$ there is one and only logical node $y \in \Gamma^-_{G^*}(x)$ of the form

$$y = \alpha x_1 x_2 .. x_{m-1}\ , \qquad (27)$$

or there are d logical nodes $y \in \Gamma^-_{G^*}(x)$ of the form

$$y = \alpha' \alpha x_1 x_2 ... x_{m-1}. \quad (28)$$

In the case of (27), there would be one and only corresponding physical node $y \in \Gamma_G^-(x)$. In the case of (28), since the split operation for node y won't take place if $|y|>|x|$, there would be two corresponding physical nodes. Thus (25) holds in this case.

2) There are no physical nodes $v \in \Gamma_G^-(x) \cup \Gamma_G^+(x)$ satisfying $|v|>|x|$. The proof for this case is similar to that for the first case, and we omit it due to lack of space.

Obviously, the in-degrees and out-degrees of all nodes in graph G satisfy

$$\sum_{x \in V(G)} \delta_G^+(x) = \sum_{x \in V(G)} \delta_G^-(x) = |E(G)|. \quad (29)$$

Thus the average in-degree of G is d. ■

The following Theorem 6 describes the upper bound for the diameter of a DL+ graph.

Theorem 6: Let graph G be a DL+ graph with base d and order N. Let the order and diameter of the initial graph G_0 be N_0 and D_0 respectively. Then $D(G)$, the diameter of graph G, satisfies

$$D(G) < 2(\log_d N - \log_d N_0 + D_0 + 1). \quad (30)$$

Proof: Let G^* be the corresponding DL graph of G with order N^*. For each physical node $x \in V(G)$, the number of logical nodes satisfies $[\![x]\!] \le d/2$. Then the order of G^* satisfies

$$N^* < \lceil d/2 \rceil \times N < d \times N. \quad (31)$$

Clearly, the path between any two nodes in G has its counterpart path with the same length in G^*. Then

$$D(G) \le D(G^*). \quad (32)$$

By (31), (32) and Theorem 4, Theorem 6 holds. ■

4. Building DHTs by using DLG technique

4.1 Self-stabilization on node joins

Initially the topology of a DLG-enabled DHT is a d-regular graph G_0. Then it evolves into a family of DL+ graphs as nodes join/leave. A new node p first look for a responsible physical node v, then update routing tables of relevant nodes. Since key assignments are usually coupled with node/object naming conventions, we will discuss them in the case study in Section 4.3.

Finding a responsible node. First, the new node p randomly generates a key and sends a query for this key, through a *gateway node n* that is already in the DHT. To find such a gateway n, node p can use any discovery mechanisms proposed in the literature. This query will eventually reach a node u' responsible for the key.

Second, new node p invokes a JOIN message from node u'. Then, at any intermediate *node u*, if u has a neighbor w satisfying $|w|<|u|$, the JOIN message will be forwarded to w; otherwise if u has a neighbor w' satisfying $(|w'|=|u|) \wedge ([\![w']\!] > [\![u]\!])$, the JOIN message will be forwarded to w'. This procedure will continue until the JOIN message reaches a physical node v which has no neighbors with shorter identifiers or more logical nodes. Node v will be *responsible* for this join event.

Updating the routing tables. This procedure is equal to take a DL+ operation ($G'=Split(G, v)$, or $G'=DL(G, v)$ followed by $G''=Merge(G', v)$) at the responsible node v. After that there would be two new physical nodes in the new DL+ graph. The first node corresponds to node v, and the second one corresponds to new node p.

4.2 Self-stabilization on node departures

Graceful departures. When a node p leaves, its routing information will be occupied by other nodes.

a) Choose a responsible node. Node p invokes a DEPART message. Then at an intermediate node u, if u has a neighbor w satisfying $|w|>|u|$, then the DEPART message is forwarded to w; otherwise if u has neighbor w' satisfying $(|w'|=|u|) \wedge ([\![w']\!] < [\![u]\!])$, this message is forwarded to w'. This process will continue until the DEPART message reaches a physical node v which has no neighbors with longer identifiers or less logical nodes.

b) Hand over logical nodes. Node v hands over its logical nodes, as well as the corresponding edges (routing table entries), to neighbor v', the logical nodes of which can be put together with the ones from v. And node p hands over its logical nodes to v. The change of the logical nodes' ownership will be noticed to all relevant neighbors. Then node p departs from the system.

c) Merge logical nodes. The number of logical nodes of v' may reach d after the handover operation from v to v'. Let the d logical nodes of v' be $x^{(i)} = \alpha_i v_1 v_2 ... v_k$, i=0,1,…, d–1. The d logical nodes will merge to one node $x = v_1 v_2 ... v_k$ with $\Gamma_{G'}^-(x) = \cup \Gamma_G^-(x^{(i)})$ and $\Gamma_{G'}^+(x) = \cup \Gamma_G^+(x^{(i)})$, where G and G' represent the graph before and after the merge operation, respectively.

Ungraceful departures. To handle an ungraceful departure, as in traditional DHTs all (physical) nodes periodically send ALIVE messages to their neighbors to detect their ungraceful departures. Once a node detects ungraceful departure of one of its neighbors, it carries through all the operations on behalf of the ungracefully departed node. The remaining process is the same.

The following Theorem 7 describes the maintenance cost of DLG-enabled DHTs per join/leave (measured in terms of the number of hops a message traverses).

Theorem 7: Let the order and diameter of the d-regular initial graph G_0 be N_0 and D_0, respectively. The cost of a join event is less than

$$3(\log_d N - \log_d N_0 + D_0) + d + 1. \quad (33a)$$

The cost of a leave event is less than

$$\log_d N - \log_d N_0 + D_0 + d - 1. \quad (33b)$$

At most $3d$ nodes need to update their routing tables.

4.3 Case study

Using the DLG technique, in order to build a DHT overlay based on a given initial graph G_0, one need

a) to describe G_0 by defining the *in-letter set* $\zeta(\alpha)$ and *out-letter set* $\psi(\alpha)$ for each node $\alpha \in G_0$; and

b) to design policies for object naming and mapping according to the notation of node identifiers.

Next we show how to build DLG-Kautz (DK), a DLG-enabled, Kautz graph-based DHT. Let $Z_d = \{0,1,\ldots, d-1\}$ be an alphabet of d letters. Define *Kautz space* as

$$KSpace(d,k) = \{x_1 x_2 \cdots x_k \,|\, x_i \in Z_{d+1}, x_i \neq x_{i+1}, i < k\}. \quad (34)$$

Let $K(d,D)$ denote Kautz graphs with diameter D and base d. Then the node set and edge set are respectively

$$V(K(d,D)) = KSpace(d,D), \quad (35a)$$

$$E(K(d,D)) = \{[x_1 x_2 \cdots x_D, x_2 \cdots x_D \alpha] \,|\, \alpha \in Z_{d+1}, \alpha \neq x_D\}. \quad (35b)$$

Examples of Kautz graphs $K(2,1)$, $K(2,2)$, and $K(2,3)$ are shown in figure 1(a), (b), and (c), respectively.

We will build DK from the initial graph $G_0 = K(d,1)$.

a) The *in-letter set* and *out-letter set* of letter α are

$$\zeta(\alpha) = \psi(\alpha) = \{\beta \,|\, \beta \in Z_{d+1}, \beta \neq \alpha\}. \quad (36)$$

b) According to (35a), the key space should be $KSpace(d, k)$ with k large enough (e.g. 100) to be able to contain all possible nodes. In [5] we have proposed a determinate algorithm (named *KautzHash*) to generate keys with base=2 which are uniformly distributed in the Kautz namespace. This algorithm could be easily extended to apply to DK to generate keys with arbitrary base, and we omit it in this paper due to lack of space.

An alternative way for mapping keys onto nodes is outlined as follows. Let $u = u_1 u_2 \ldots u_m$, $s = s_1 s_2 \ldots s_r$. Define $P(u,s)$ as the maximum value of all integers i, $0 \leq i \leq \min(m,r)$, which satisfy $u_{m-i+j} = s_j$ for any j with $1 \leq j \leq i$. A key s is mapped onto a node u, iff (37) holds.

$$P(u,s) = |u|; \text{ or } P(u,s) = |u| - 1 \text{ and } u_1 = s_r. \quad (37)$$

5. Evaluation

We utilize DK as a model to evaluate the DLG technique. We first evaluate the diameter and average routing path length of DK (d=4, d=16), and compare it with other constant-degree DHTs including CAN and Koorde. Queries are generated randomly at a node, destined for keys uniformly distributed in the key space.

The results are plotted in Figure 6. The diameter and average path length of DK are denoted as DK (avg) and DK (max), respectively. Note although FissionE can only initiate from Kautz graph $K(2,1)$, we show it in the first figure for comparison. From Figure 6 we can infer that both the diameter and average path length of DK are much less than those of other DHTs. The difference is more pronounced at a larger network size.

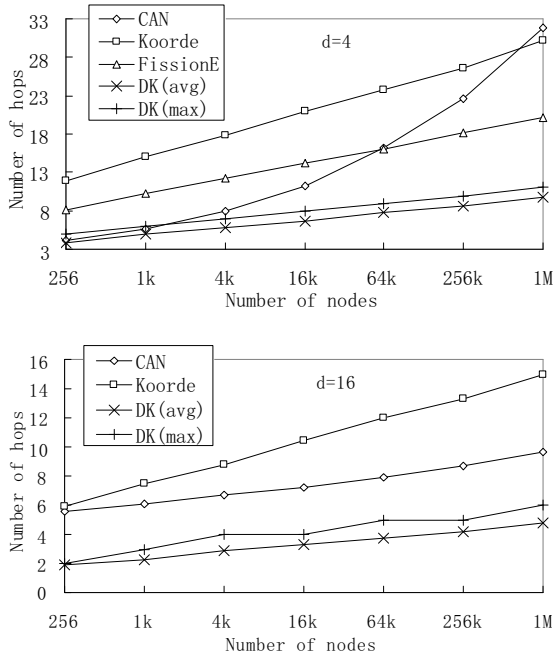


Figure 6. **Diameter and average path length**

Figure 7 shows the probability distribution of routing path lengths of DK (d=4, d=16) for a network size of 1 million nodes. The low variance indicates that the lengths of most paths are very close to the mean.

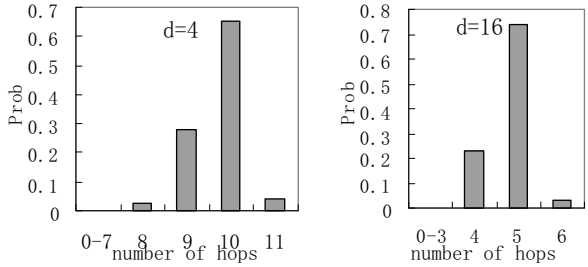


Figure 7. **Path length distribution**

We then evaluate the average message cost per join/leave event in DK with d=16 for a network size of 1 million. There are 100 gateway nodes and a new node randomly chooses one to start the join procedure. The results are shown in Figure 8(a), which also includes the cost in CAN for comparison. We conclude that the cost of a join event in DK is much less than that in CAN. Note that the cost of a leave event in CAN is 0, which is because CAN utilizes an asynchronous background process to reassign its zones.

We then evaluate the distribution of node identifier lengths in DK with d=16 and network size of 1 million.

The results are shown in Figure 8(b). More than 70% nodes have the same identifier length of 5 and no nodes have an identifier length more than 6. We conclude that the distribution of identifier lengths in DK is almost uniform and DK has a good load balance property.

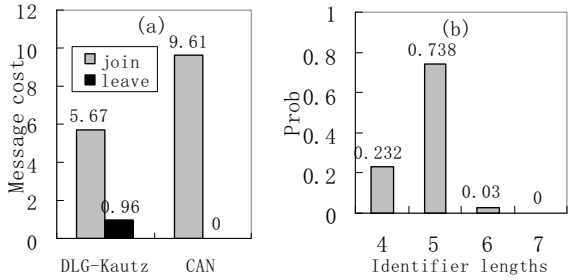


Figure 8. **Message cost & ID length distribution**

6. Related work

CAN [1] uses a d-dimensional Cartesian coordinate space (for some fixed d) and its degree is $2d$. Each node in CAN corresponds to a zone. Zones split or merge when nodes join/leave. It utilizes a bit-correct routing algorithm. The diameter of CAN is $1/2dN^{1/d}$.

Koorde [4] is based on the de Bruijn graphs and achieves a diameter of $O(\log N)$ with 2 neighbors per node. Koorde uses the immediate real predecessor to simulate nodes that are not active on the ring. However, these bounds are achieved only in the expectation and with a constant in the "big O" notation.

FissionE [5] utilizes Kautz graph $K(2,D)$ as its initial topology. It has an average degree of 4 and a diameter of $O(\log_2 N)$. It uses a space partitioning method to add new nodes to the overlay. Its routing algorithm is similar to that in static Kautz graphs. Since the base can only be 2, it cannot achieve better performance with higher bases.

Moore [6] uses the PLG technique [8] and proposes the *incomplete Kautz digraph* to design a P2P network based on Kautz graphs with arbitrary base. As discussed in [12], however, Moore requires *centralized* control and global information for its topology maintenance, and thus is not a practical P2P scheme.

To address the problem of Moore, SKY [12] uses a decentralized technique called *distributed Kautz graphs* (DKG) for its topology maintenance, which borrows the idea of edge-node transition from line graphs to get a series of P2P topologies based on Kautz graphs. SKY is the first effective P2P network based on Kautz graphs with any base. Clearly, DKG is a customized technique which can only be applied to Kautz graphs, while DLG is a universal framework which could be applied to arbitrary constant-degree graphs. In a nutshell, DLG is a generalization of DKG, and DKG could be viewed as a specialization of DLG in Kautz graphs.

7. Conclusion

In this paper we present the design of distributed line graphs, a universal framework for building DHTs based on arbitrary constant-degree graphs. We show the power of DLG technique by applying it to Kautz graphs to build an efficient DHT named DK.

In the future we will utilize the DLG technique to make an in-depth study to gain a deeper insight of common design choices of constant-degree DHTs. We also plan to implement a prototype of the DK DHT.

Acknowledgement

This work is sponsored in part by the National Basic Research Program of China (973) under Grant No. 2005CB321801, National Natural Science Foundation of China under Grant No. 60673167 and 60703072. This work is also partially sponsored by grants from NSF CISE CSR program, CyberTrust Program, an IBM SUR Grant, and an IBM Faculty Award.

References

[1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker. A Scalable Content Addressable Network. *SIGCOMM* 2001.

[2] D. Malkhi, M. Naor, D. Ratajczak. Viceroy: A Scalable and Dynamic Emulation of the Butterfly. *PODC* 2002.

[3] P. Fraigniaud and P. Gauron. D2B: A de Bruijn Based Content-Addressable Network. *Theoretical Computer Science*, 355(1):65–79, 2006.

[4] F. Kaashoek and D. Karger. Koorde: A Simple Degree-optimal Distributed Hash Table. *IPTPS* 2003.

[5] D. Li, X. Lu, J. Wu. FISSIONE: A Scalable Constant-degree and Low Congestion DHT Scheme Based on Kautz Graphs. *INFOCOM* 2005.

[6] D. Guo, J. Wu, H. Chen, X. Luo. Moore: An Extendable Peer-to-Peer Network Based on Incomplete Kautz Digraph with Constant Degree. *INFOCOM* 2007.

[7] J. Xu, A. Kumar, X. Yu. On the Fundamental Tradeoffs between Routing Table Size and Network Diameter in Peer-to-Peer Networks. *JSAC*, 22(1):151–163, 2004.

[8] M. A. Fiol and A. S. Llado. The Partial Line Digraph Technique in the Design of Large Interconnection Networks. *Transactions on Computers*, C-41(7):848–857, 1992.

[9] P.Tvrdik. Kautz Necklaces. *Research Report* 94-08, LIP ENSL, France, Mar. 1994.

[10] P.Tvrdik. Factoring and Scaling Kautz Digraphs. *Research Report* 94-15, LIP ENSL, France, Apr. 1994.

[11] D Du, Y. Lyuu and D. F. Hsu. Line Digraph Iterations and Connectivity Analysis of de Bruijn and Kautz Graphs. *Transactions on Computers*, 42(5):612–616, 1993.

[12] Y. Zhang. SKY: Efficient Peer-to-Peer Networks Based on Distributed Kautz Graphs. *Research Report*, NUDT, Aug. 2007. http://www.kylinx.com/Papers/SKY.pdf.