Improving the Scalability of Logarithmic-Degree DHT-based Peer-to-Peer Networks

Bruno Carton
CETIC
8 rue Clément Ader
B-6041 Gosselies
bruno.carton@cetic.be

Valentin Mesaros
Univ. catholique de Louvain
2 place Sainte Barbe
B-1348 Louvain-la-Neuve
valentin@info.ucl.ac.be

Peter Van Roy
Univ. catholique de Louvain
2 place Sainte Barbe
B-1348 Louvain-la-Neuve
pvr@info.ucl.ac.be

Report No. UCL/INFO-2004-01
January, 2004

Abstract

High scalability in Peer-to-Peer (P2P) systems has been achieved with the emergence of the networks based on Distributed Hash Tables (DHT). Most of the DHTs can be regarded as exponential networks. Their network size evolves exponentially while the minimal distance between two nodes as well as the routing table size, i.e., the degree, at each node evolve linearly or remain constant. In this paper we present a model to better characterize most of the current logarithmic-degree DHTs. We express them in terms of *absolute* and *relative* exponential structured networks. In relative exponential networks, such as Chord, where all nodes are reachable in at most H hops, the number of paths of length inferior or equal to H between two nodes grows exponentially with the network size. We propose the `Tango` approach to reduce this redundancy and to improve other properties such as reducing the lookup path length. We analyze `Tango` and show that it is more scalable than the current logarithmic-degree DHTs. Given its scalability and structuring flexibility, we chose `Tango` to be the algorithm underlying our P2P middleware.

1 Introduction

Over the past few years, Peer-to-Peer (P2P) networks have become an important research topic due to their interesting characteristics/potentials such as self-organization, decentralization and scalability. The underlying principle relies on exchanging data only between the interested parties. A P2P network is principally characterized by its structuring policy and the lookup protocol employed.

Not long after the emergence of the first popular P2P networks, Napster and Gnutella, it was realized that scalability in these networks was an important issue. P2P networks based on DHT (Distributed Hash Table) have been proposed as a solution. These networks are self-organized, fully distributed and highly scalable. Furthermore, given that each node has a well defined routing table, the lookup for any node/item can be accomplished within a relatively small number of hops. As the network size increases *exponentially*, the maximum lookup length as well as the routing table size at each node (i.e., the degree) increase *linearly* like in Chord [1], Pastry [2] and Tapestry [3], or even remain constant like in Koorde [4] and Distance Halving (DH) [5].

The DHT based P2P networks are also called *structured networks*, since they follow a well defined structure. A closer look to their structure allowed us to notice that most of the logarithmic-degree DHTs fall into two main categories, depending on the nodes' view of the network (we defer the definition of node's view to Section 2). We call them *absolute* and *relative structured exponential networks*.

A first contribution of this paper is the description of a model to better characterize the exponential structured networks as absolute and relative. Related to this work is the research described in [6] where a model based on the concept of k-ary search is proposed for reasoning about DHT networks. Their model addresses only relative structured exponential networks, while ours is more general, addressing the absolute networks, too.

Our model allowed us to observe that in the relative exponential structured networks the fingers of a node are not totally exploited. Hereinafter we denote the "fingers" of a node n to be the single-hop connections of n, and hence representing the entries in the routing table of node n. In Section 3 we propose an approach, that we called `Tango`, to structure the relative exponential networks for increasing their scalability. `Tango` reduces the redundancy in the multiplicity of paths between two nodes of a relative exponential network and, as such, it reduces the path length between the nodes. The `Tango` approach is the second and the main contribution of this paper. In Section 4 we compare `Tango` with DKS [7] and with the DH constant-degree network. Note that DKS generalizes Chord to allow a tradeoff between the maximum lookup length in the network and the size of the routing table at each node (e.g., for an arity of 2, DKS has the same structure as Chord).

The remainder of the paper is organized as follows. We continue with the description of our model for absolute and relative logarithmic-degree structured exponential networks. In Section 3 we present the `Tango` approach and its routing policy. In Section 4 we analyze the scalability, the path overlap and the symmetry support in `Tango` and then we conclude.

2 Structured exponential network

In the following we explain the general building rule of a structured exponential network and we identify two kinds of networks: the absolute and the relative exponential network. In order to obtain a more readable description, we use the following notations: Net_i stands for the exponential network corresponding to the i^{th} incremental building step, and k stands for the exponential factor.

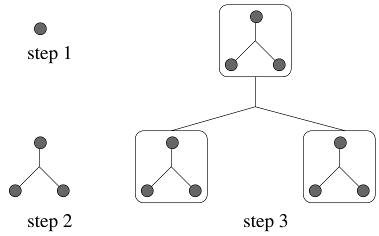

Figure 1: The first 3 steps of an exponential network with $k = 3$.

2.1 Building rules

A structured exponential network is a network built incrementally using well-defined steps. It is composed of nodes linked together via directed edges according to structuring rules, and characterized by an exponential factor k which is the number of instances of network Net_i used to define the subsequent network Net_{i+1}. As shown in Figure 1, the network Net_1 is the initial network composed of one node. At step i, network Net_i is built by using k instances of network Net_{i-1} linked to one another. Given this building pattern, one can observe that the network size grows exponentially with the number of steps while the maximum number of steps (say hops) needed to reach a node grows linearly with the number of steps.

We identify two methods for connecting all k instances of Net_{i-1} at the i^{th} step : absolute and the relative connections (i.e., fingers). They lead to absolute and relative structured exponential network, respectively. We illustrate both methods for a network of size 64, built in four steps, and parameterized by an exponential factor $k = 4$. Each node is identified both numerically by using a unique identifier ranging from 0 to 63, and graphically by using k shapes (i.e., light square, light circle, bold square and bold circle). The shape organizes the nodes in the network whereas the size of the shape determines the network building step. Small shapes stand for instances of Net_1, medium shapes for instances of Net_2, and large shapes for instances of Net_3. The network instance of Net_4 regroups the four network instances of Net_3. However, for simplicity, Net_4 is not marked in the figures. In order to distinguish the fingers of the reference node from the other nodes, we represent the fingers as non-gray numbers whereas the other nodes are in gray. Moreover, we introduce the $\oplus$ and the $\ominus$ operators. In a network of size S, we define both operators as $m \oplus n = (m + n) \bmod S$ and $m \ominus n = (m - n + S) \bmod S$.

2.2 Absolute structured exponential network

An absolute structured exponential network is represented in Figure 2 (left). In such a network, each node has the same view of the network. For instance, all nodes see that nodes ranging from 0 to 15 are sitting in the large light square while nodes ranging from 48 to 51 are sitting in the medium light square inside the large bold square. That is, if a node sees that a node m is sitting in a given shape then all the nodes see that m is sitting in that given shape. In such a network, at the i^{th} step, the $k - 1$ fingers of a node n are

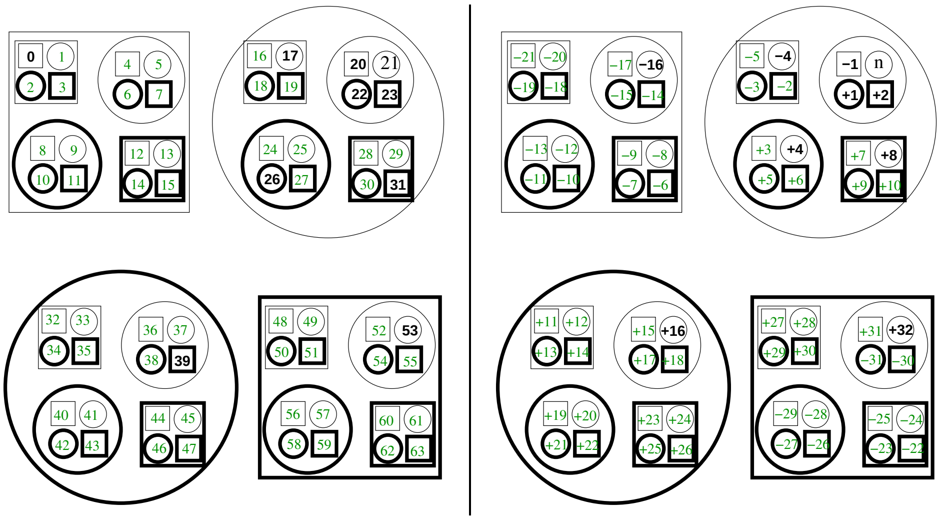

Figure 2: (left) Absolute structured exponential network of size 64 with $k = 4$. (right) View of a node n in a relative structured exponential network of size 64 with $k = 4$.

pointing to the $k - 1$ other instances of Net_{i-1}. Moreover, as long as there is a finger of n pointing to each Net_{i-1}, it does not matter to which node inside each Net_{i-1} n points to. For instance, in the network represented in Figure 2 (left), the fingers of node 21 at the third step can be any instance of nodes a,b,c where $a \in [16 \ldots 19]$, $b \in [24 \ldots 27]$, and $c \in [28 \ldots 31]$.

2.3 Relative structured exponential network

A relative structured exponential network differs from an absolute one by the fact that the view of the network owned by a particular node is relative to its position winthin the network. In Figure 2 (right) we express graphically the relative approach where we reuse the same graphical notations as in Figure 2 (left) for a network of size 64 and $k = 4$. We represent the view of a node n, and all the other nodes are denoted relatively to n. That is, a node denoted as $+m$ corresponds to $n \oplus m$ and a node denoted as $-m$ corresponds to $n \ominus m$. In such a network, nodes sitting in the large light square are found at distance $dist$ from the reference node, with $-21 \leq dist \leq -6$. As two examples, we consider first $n = 21$ and then $n = 29$. For n equal 21, the large light square contains nodes ranging from 0 to 15. On the other hand, for n equal 29, the large light square contains nodes ranging from 8 to 23. Moreover, in a relative exponential network, a node n has to point precisely to the nodes occupying relatively the same positions in the $k - 1$ other instances of Net_{i-1}. For instance, from Figure 2 (right) one can note that the fingers of nodes 29 and 21 are those represented in Table 1.

Table 1: Fingers of nodes 29 and 21 for an instance of Net_4 characterized by k= 4 of a relative structured exponential network.

Step	Offset	Fingers of nodes	
		29	21
2	-1	28	20
	$+1$	30	22
	$+2$	31	23
3	-4	25	17
	$+4$	33	25
	$+8$	37	29
4	-16	13	5
	$+16$	45	37
	$+32$	61	53

2.4 Relation between DHT and exponential networks

Most logarithmic-degree DHT-based P2P networks can be expressed either in terms of an absolute or in terms of a relative structured exponential network. For instance, Pastry and Tapestry can be seen as instances of the absolute structured exponential network by instantiating the employed alphabet to the shapes used in Figure 2. On the other hand, Chord and DKS can be seen as instances of the relative structured exponential network. For example, as shown in Figure 2 (right), considering that the position of a node n is the small light square, inside the medium light square, inside the large light square, n points to nodes found at distances 1,2,3; 4,8,12; 16,32,48 which correspond exactly to the distances at which a node must point in a DKS network characterized by an arity of $k = 4$.

This model allows us to state that networks built with the relative and the absolute approaches scale at the same rate. Indeed, let S_i be the size of network Net_i and H_i be the maximum number of hops to reach any node in Net_i. Then, for both structures we have $S_i = k * S_{i-1}$ with $S_1 = 1$, $H_i = i - 1$, and a number of $(k-1) * (i-1)$ fingers at each node.

3 `Tango`: a novel approach for reducing unexploited redundancy

In a relative exponential network we can identify two types of redundancy. The first one results from the commutative property of the addition operation and from the fact that each node owns the same fingers relatively to its position. For example, in Chord, node 0 can reach node 6 via node 4 (6=0+4+2) and also via node 2 (6=0+2+4). The second type of redundancy results from the underutilization of fingers.

To have a clear explanation, we introduce the notion of positive and negative regions of a given node n. A node m is found in the positive region of node n iff $m \ominus n < n \ominus m$, otherwise, node m is found in the negative region of node n.

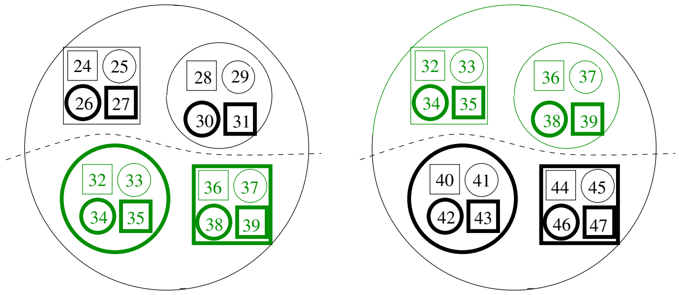

Figure 3: The areas reachable by node 21 in at most 3 hops via node 29 and node 37 respectively. Both areas overlap on region $[32 \ldots 39]$.

We propose `Tango`, an approach to address the second type of redundancy, and thus increasing network scalability. The region covered in at most i hops via the farthest finger added in the positive (resp. negative) region at step i and the region covered in at most i hops via the closest finger added in the positive (resp. negative) region at step $i+1$ overlap partially. For example, as shown in Figure 3, the regions reachable in at most 3 hops by node 21 via node 29 (i.e., from 24 to 39) and via node 37 (i.e., from 32 to 47) overlap. Let a valid path between two nodes in a network instance Net_i be any path between these nodes whose length is at most $i-1$ hops. In a relative network, all these overlap regions increase exponentially the number of valid paths between two nodes. Moreover, the cumulated size of the overlap, i.e., the amount of unexploited redundancy in an instance of Net_i grows exponentially with i.

3.1 `Tango` definition

In order to prevent overlapping, the region comprised between the farthest finger added in the positive (resp. negative) region at step i and the closest finger added in the positive (resp. negative) region at step $i+1$ has to be equal to the size of the network instance Net_i. This represents the key idea behind the `Tango` approach. This improvement can be done at each building step. It is graphically expressed in Figure 4 for a network characterized by $k = 5$. One can notice that Net_{i+1} is composed of 5 blocks. There are 4 instances of Net_i and 1 instance of $Extended\ Net_i$, which is the network reachable by the reference node in at most i hops by using the fingers defined in Net_i.

Let k_i^+ and k_i^- be the number of fingers added in the positive and ,repsectively, the negative regions of a node at step i. We can put k_i^+ and k_i^- in relation with the exponential factor k of the network. Hence, knowing that at each construction step i there are $k-1$ fingers added to a node, we obtain $k = k_i^+ + k_i^- + 1$.

Let $d_{i,j}^+$ (resp. $d_{i,j}^-$) be the distance at which the j^{th} positive (resp. negative) finger of the i^{th} step should be placed. Let S_i^+ (resp. S_i^-) be the size of the positive (resp. negative) region of a reference node at step i. By remembering that S_i is the network size at step i, one can establish the equations characterizing the size growth and the fingers positioning in `Tango` as shown in Equations 1. One can note that for $k_i^- = 0$,

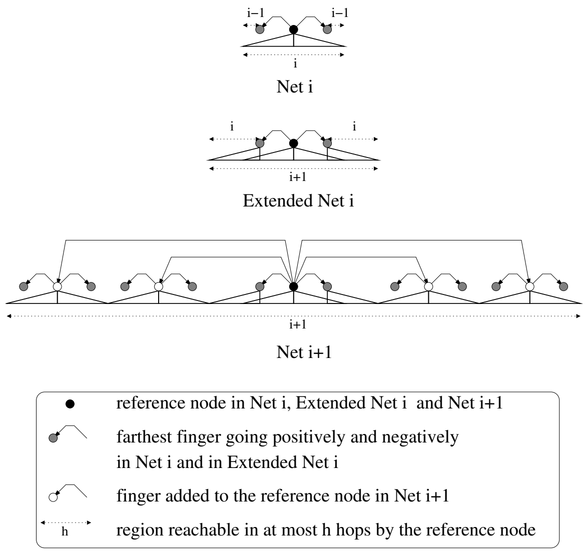

Figure 4: Network building pattern in `Tango` where $k = 5$.

the `Tango` network corresponds to an improved version of Chord, and DKS with an arity k. The reader can refer to Section 4 for a comparison between Chord, DKS and `Tango`.

$$\begin{array}{llll}
d^{\pm}_{2,j} = j & j \in [1 \dots k^{\pm}_{2}] & S^{\pm}_{1} = 0 & \\
d^{\pm}_{i,j} = d^{\pm}_{i,j-1} \pm S_{i-1} & j \in [1 \dots k^{\pm}_{i}],\ i > 2 & S^{\pm}_{i} = S^{\pm}_{i-1} + d^{\pm}_{i,k^{\pm}_{i}} & i > 1 \\
d^{\pm}_{i,0} = d^{\pm}_{i-1,k^{\pm}_{i-1}} & i > 2 & S_{i} = S^{+}_{i} + S^{-}_{i} + 1 & i > 0
\end{array} \tag{1}$$

3.2 Key-based routing

The purpose of key-based routing is to map a key to a node. Thus, given a message tagged with key Key, the routing mechanism consists in forwarding the message from the sender to the node responsible of Key. Let p^{+}_{n} (resp. p^{-}_{n}) be the first node encountered in the positive (resp. negative) region of n. The responsibility of a node n is defined in Equation 2.

Beside the node responsibility, there is the finger responsibility defining the node to which a message should be forwarded to. In `Tango` we split the finger responsibility of a given finger F in negative and positive sides[1]. Than, let the focused network be an instance Net_l and let $Sp^{\pm}_{i,j}$ (resp. $Sn^{\pm}_{i,j}$) be the sizes of the positive (resp. negative) finger responsibility as defined in Equations 3 and 4. The finger responsibility $R^{\pm}_{i,j}$

[1]The denomination of `Tango` comes from its ability to have positive routing steps followed by negative routing steps and vice versa.

of finger located at position $P^{\pm}_{i,j}$ related to the distance $d^{\pm}_{i,j}$ are defined in Equation 5. Hence, by using its finger $F_{i,j}$, a node can cover the region $R_{i,j}$ in at most $i-1$ hops. One can note that for $k_2^- = 0$, the finger responsibility defined in Equation 5 correspond to the one employed in Chord and DKS.

$$R_n = \left[n \ominus \left(\left\lfloor \frac{k_2^+}{k-1} \right\rfloor * (n \ominus p_n^- \ominus 1) \right) \ldots \left(\left\lceil \frac{k_2^-}{k-1} \right\rceil * (p_n^+ \ominus n \ominus 1) \right) \oplus n \right] \quad (2)$$

$$Sp^-_{i,j} = S^-_i \; ; \; Sp^+_{l,k_l^+} = S^+_i \; ; \; Sp^+_{i,k_i^+} = S^+_{i+1} \; ; \; Sp^+_{i,j} = S^+_i \quad (3)$$

$$Sn^+_{i,j} = S^+_i \; ; \; Sn^-_{l,k_l^-} = S^-_i \; ; \; Sn^-_{i,k_i^-} = S^-_{i+1} \; ; \; Sn^-_{i,j} = S^-_i \quad (4)$$

$$R^{\pm}_{i,j} = \left[P^{\pm}_{i,j} \ominus Sn^{\pm}_{i,j} \ldots P^{\pm}_{i,j} \oplus Sp^{\pm}_{i,j} \right] \quad (5)$$

3.3 `Tango` in a sparse and dynamic network

In a sparse network, the position of a finger F (i.e., P) of a node n may correspond to a missing node. In that case, n points to the node responsible of P. Hence, the nodes are playing the finger role of the missing nodes laying within their responsibility. In order to preserve the lookup efficiency, each node adapts its routing table in order to reach the same part of the network in the same number of hops as it would have been done by each missing nodes within its responsibility. That is why in `Tango` we define the finger position $P(n)$ and the finger node $F(n)$ of a node n as in Equation 6, where $j \in [1 \ldots k_i^-]$, $g \in [1 \ldots k_i^+]$ and $i \in [1..l]$.

$$\begin{array}{rclcrcl} P^-_{i,j}(n) & = & R_n.inf \ominus d^-_{i,j} & & F^-_{i,j}(n) & = & m \quad s.t. \quad P^-_{i,j}(n) \in R_m \\ P^+_{i,g}(n) & = & R_n.sup \oplus d^+_{i,g} & & F^+_{i,g}(n) & = & m \quad s.t. \quad P^+_{i,g}(n) \in R_m \end{array} \quad (6)$$

To deal with the dynamics in a `Tango` network, the algorithms of join, fault tolerance and correction on use defined in DKS can be applied directly to `Tango`. Moreover, due to the symmetry provided by the `Tango` networks featured with $k^+ = k^-$, the correction on use can be made more efficient.

4 Analysis

In this section we compare `Tango` with DKS and Chord, and shortly with the Distance Halving constant-degree network. Unless stated otherwise, in our analysis we consider fully populated networks.

4.1 `Tango` vs. DKS

DKS generalizes Chord to allow a tradeoff between the maximum lookup length in the network (i.e., the diameter) and the size of the routing table at each node (i.e., the degree). The structure of DKS characterized by $k = 2$ is the same as the one of

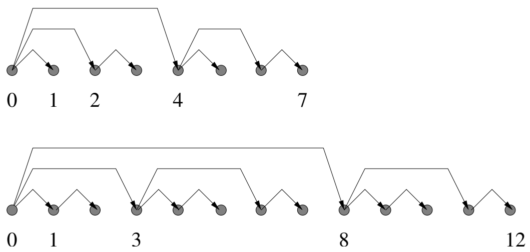

Figure 5: Paths from node 0 to all the other nodes in a Chord network of size 8 and in a `Tango` network of size 13.

Chord. `Tango` also supports the tradeoff between the diameter and the degree. In this subsection we analyze the scalability, the path overlap and the routing symmetry of both systems.

4.1.1 Scalability

The graphical description of the `Tango` network building pattern (Figure 4) highlights qualitatively the improvement of the network scalability. One can be interested in a more quantitative analysis. In this subsection we show that the size of the network covered with `Tango` is much larger than the size of the network covered with DKS and Chord, while keeping the same network diameter and the same degree at a node. That is, in `Tango` the exponential factor is bigger than in Chord and DKS. Recall that for $k = 2$, DKS is equivalent to Chord. From the size growth Equation 1, one can deduce the size of the network (Equation 7) covered with `Tango` at a step $i > 2$.

$$S_i = (k+1) * S_{i-1} - S_{i-2} \tag{7}$$

$$z_1 = \frac{k+1+\sqrt{(k+1)^2-4}}{2}\ , \quad z_2 = \frac{k+1-\sqrt{(k+1)^2-4}}{2} \tag{8}$$

The solutions of the characteristic equation corresponding to Equation 7 are z_1 and z_2; see Equation 8. They are used to express S_i in the non-recursive Equation 9.

$$S_i = \frac{(z_1 - 1) * {z_1}^{i-1}}{z_1 - z_2} - \frac{(z_2 - 1) * {z_2}^{i-1}}{z_1 - z_2} \tag{9}$$

For a step i sufficiently high, the second term of Equation 9 can be ignored. Thus, the growth ratio, i.e., the exponential factor, of the network size in `Tango` is equivalent to z_1, where $z_1 > k$. This allows us to approximate the network size covered with `Tango` at a step $i > 1$ by $S_i = {z_1}^{i-1}$, while the network diameter and the degree at a node are the same as in DKS, i.e., $H_i = i - 1$ and $(k-1) * (i-1)$, respectively.

The `Tango` approach of network construction (Figure 4) and the corresponding equations (Equation 1) provide us with another way to compare the size growth in `Tango` and DKS. That is, Equation 10 represents the network size covered by `Tango` at the i^{th} construction step.

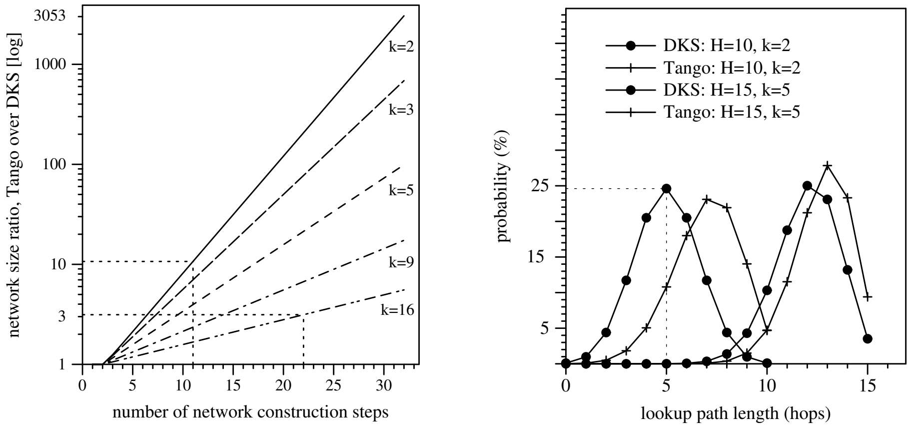

Figure 6: (left) Ratio between the network sizes covered by Tango and DKS, with the same number of fingers at given construction steps. (right) Distribution of the shortest path in Tango and DKS for k=2;5 and diameter equal to 10 and 15, respectively.

$$S_i = k^{i-1} + \sum_{j=1}^{i-2} k^{i-j-2} * (d^{+}_{j,k^{+}_{j}} + d^{-}_{j,k^{-}_{j}}) \qquad i > 2 \tag{10}$$

One can note that the first term of Equation 10 corresponds to the network size covered by DKS at the i^{th} step, i.e., k^{i-1}. The second term, which also increases exponentially, corresponds to the difference between the two network sizes. It represents the cumulated unexploited redundancy in DKS. In Figure 5 we present an example of how Tango covers a larger network than Chord (DKS, $k = 2$) even at the very early building steps. With a routing table of size 3, a node in Chord (DKS, $k = 2$), node 0 in our example, can cover a network of size 8 in 3 hops. On the other hand, in Tango, in 3 hops, a node can cover a larger network, i.e., of size 13.

To better understand the relation between Tango and DKS, in Figure 6 (left) we plotted the ratio between the network sizes covered in Tango and DKS at each construction step ranging from 1 to 32, for five different values of k. Recall that the network construction step corresponds to the maximum number of hops to reach any node in the corresponding network, and that the number of fingers added at each step is the same in both Tango and DKS networks, i.e., $k - 1$. One can note two things here. First, for a given k, the ratio between the network sizes is growing exponentially at each step. For instance, for $k = 2$ (the case of Chord), one can see that at step 11 the size of the network covered by Tango is 10 times larger than the size of the network covered by DKS, while at steps 16 and 32 the ratio is more than 20 and 3053 respectively. Second, it is interesting to note that the growth ratio of the ratio decreases as k increases. For instance, for $k = 16$ the ratio equals 3 only at step 22. However, since increasing k in a P2P system leads to increasing the resource consuming, e.g., routing table size and the associated maintenance messages, at very each node in the network, it is likely that relative small values of k will be employed.

We were also interested in the distribution of the shortest path in Tango with respect to the one in DKS (and implicitly the one in Chord). In Figure 6 (right) we show this distribution for $k = 2$ and $k = 5$, and network diameter equal to 10 and 15, respectively. One can note that, while for DKS we get a proper bell-shaped distribution, for `Tango` we get a close distribution, but translated to the right. This tells us that, while in DKS the proportion short paths/long paths is equal, in `Tango` there are more long paths than short paths.

4.1.2 Path overlap

As indicated in [8], node-disjoin paths are attractive to P2P networks since they provide independent backup routing options when the main shortest path fails. They also show that de Bruijn networks offer much more paths that do not overlap than Chord. We were interested to see what is the path overlapping in `Tango`. For this end, in the following, we recall the graph definitions from [8].

Define $P(x, y)$ as the set of all vertices along some path x to y. Denote $Q(x, y)$ to be the set of all vertices in $P(x, y)$, except x and y: $Q(x, y) = P(x, y) \setminus \{x \cup y\}$. For any pair of nodes (x, y), define $P_i(x, y)$ to be the shortest path (according with the greedy routing rules of the corresponding graph) to y through x's neighbor i. Define $T(x, y) = \sum |P_i(x, y)|$ to be the total number of vertices in all shortest paths $P_i(x, y)$, and $U(x, y) = | \cup \ Q_i(x, y)|$ to be the number of unique vertices[1] in all such paths. Then, define the average percentage of unique nodes in all parallel paths as in Equation 11, and the *path overlap* to be $J(G) = 1 - U(G)$.

$$U(G) = \frac{\sum\limits_x \sum\limits_y U(x, y)}{\sum\limits_x \sum\limits_y T(x, y)} \tag{11}$$

We computed the path overlap for Chord (DKS, $k = 2$) and `Tango` (the asymmetric construction, $k = 2$), where we considered close values of the network size. Thus, we chose S_{14} and S_{15} for Chord, and S_{10} for `Tango`. The corresponding values for the path overlap are 0.3909 and 0.3981, and 0.3985 respectively. One can see that the path overlap in both Chord and `Tango` is quite similar.

Then, we were interested in the number of non-overlapping paths $Q_j(x, y)$ between any pair of two nodes x and y in the network. We exclude pairs like (x, x) and pairs between a node and its fingers. In Figure 7 we show the distribution of the number of non-overlapping shortest paths in DKS and `Tango` for networks corresponding to $k = 2$ and to the 10^{th} iteration of the network construction step S_{10}, i.e., size of 1024 and of 10946, respectively. One can note that for DKS there are 50% of pairs for which there are two non-overlapping shortest paths. The percentage of pairs having more non-overlapping paths is decreasing as the number of paths increases.

In the case of the `Tango` network the distribution is quite different. First, note that there is a significant percentage of pairs (23$\%$) with only one non-overlapping path. That is, for these pairs of nodes *all* the alternative shortest paths have some nodes in common with the best shortest path. Second, while in DKS the number of

[1] Note that the definition of the number of *unique* vertices as provided in [8] contains a small error. They defined it as being $U(x, y) = | \cup P_i(x, y)|$.

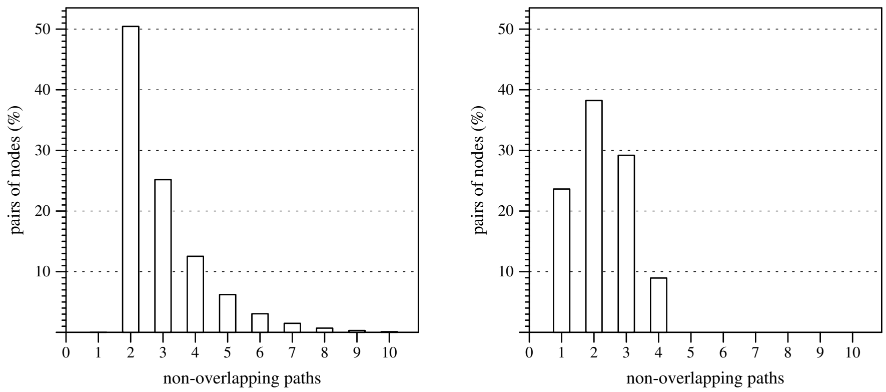

Figure 7: Distribution of the number of non-overlapping shortest paths in DKS (left) and `Tango` (right) for S_{10} with $k = 2$.

non-overlapping paths varies from two to the network diameter, in `Tango` the number of non-overlapping paths between two nodes is maximum four. Moreover, we conjecture that in `Tango` networks of S_i with $i > 3$ and $k = 2$ the maximum number of non-overlapping paths between any two nodes is maximum four. One can increase the number of non-overlapping paths in `Tango` by increasing the path redundancy. Thus, one can modify Equation 1 to obtain a tradeoff between the path redundancy and scalability.

4.1.3 Symmetry

In general, a network has different characteristics, but the way it is organized together with the corresponding routing policies are of great importance. A DHT based P2P network organizes itself such that it achieves efficient routing while being scalable with the network increase, and resilient to individual node failures.

Within a network, each node has a well determined set of fingers which constitute the routing table of that node. The rules for organizing the routing tables in a network characterize the entire network. Two interesting properties of a P2P network are: the *routing entry symmetry* and the *routing cost symmetry*.

Routing entry symmetry. This symmetry provides the property that for any two different nodes of a network, n and p, if p has a finger to n, then n also has a finger to p (or in the neighborhood of p). This symmetry provides a node with the ability to perform in-place notifications of routing entry changes.

Routing cost symmetry. The routing entry symmetry and the associated lookup protocol provide a network with another property: the routing cost symmetry. That is, it is very likely that a lookup from a node n to another node p takes the same number of hops as a lookup from p to n.

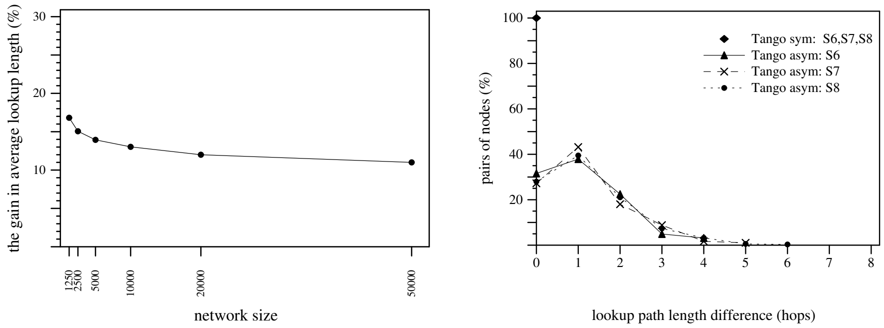

Figure 8: (left) The gain in average lookup length of symmetric, poorly populated vs. asymmetric networks, with $k = 3$. (right) Distance variation between pairs of nodes in symmetric and asymmetric `Tango` for three fully populated networks corresponding to S_6, S_7 and S_8.

These properties can be obtained by using a symmetric approach for organizing the routing table of the nodes within a network. This approach was successfully applied in Hyperchord [9] and S-Chord [10] P2P systems. The routing entry symmetry property can be exploited to diminish the number of message exchanged for maintaining the routing tables.

The `Tango` structuring approach proposed in the Section 2 allows one to organize routing tables in order to achieve routing entry symmetry and routing cost symmetry. To construct a `Tango` network characterized by the two routing symmetry properties, one have to (i) choose an odd exponential factor ($k = 3, 5, 7, ...$), and (ii) have the same number of connections in the negative hand side as in the positive hand side of a node, i.e., $k_i^- = k_i^+$, added at each construction step i. One can see that as the difference between k_i^- and k_i^+ increases, the symmetry in the network decreases. For instance, when $k_i^+ = k - 1$ we have an asymmetric network similar to Chord and DKS.

As already showed in Section 3, the `Tango` network scales similarly, regardless whether it is symmetric or asymmetric. However, in our research, we were also interested in looking at the average lookup length in both types of network. We did some measurements on highly sparse networks of different sizes. We considered networks with $k = 3$ and the identifier space of size of approximately $1.117 * 10^9$, which corresponds to the 17^{th} size iteration when constructing the network, S_{17}. The network nodes were randomly chosen. We considered the query distribution to be uniform over the identifier space. In Figure 8 (left) we show the gain in average lookup length of the symmetric, poorly populated network with respect to the asymmetric one. One can see that in the symmetric networks the average lookup length is smaller than in the asymmetric ones. As the network becomes more populated, the gain decreases. This is normal since in a fully populated network the average lookup length is the same for both types of networks.

We continued the measurements with analyzing the routing cost symmetry for the symmetric and asymmetric approaches in fully populated networks. We measured

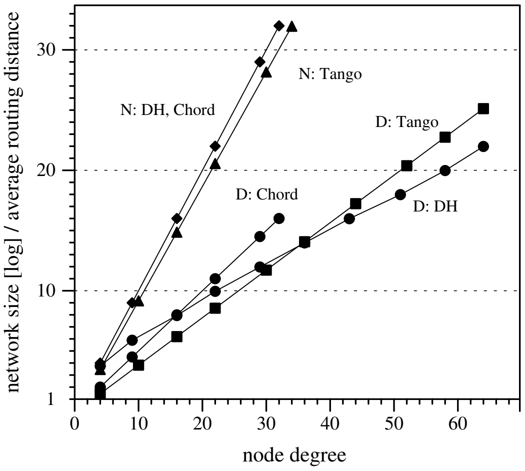

Figure 9: Network size [`N`] and average routing distance [`D`] of `Tango`, DH and Chord, with respect to different values of node degree.

the percentage $P(x)$ of any pair of two nodes n and p such that the absolute difference between the distance (n, p), in number of hops, and the distance (p, n), equals $x = |H(n \rightarrow p) - H(p \rightarrow n)|$. For this test, we considered $k = 3$ and the identifier spaces of different sizes, corresponding to S_6, S_7 and S_8. As illustrated in Figure 8 (right), the symmetric approach provides full symmetry for the routing cost, i.e., for 100% of pairs the difference is 0. In contrast, with the asymmetric approach the percentage of pairs with difference equal to 0 is less than 35% for all three studied networks. Furthermore, we have around 40% of pairs with difference equal to 1. One can note that with the asymmetric approach we have pairs characterized by a strong asymmetry of the routing cost.

4.2 `Tango` vs. constant-degree networks

A constant-degree network is a network whose size can increase exponentially, while the node degree remains fixed and the diameter increases logarithmically. Some examples are those based on the de Bruijn graph, such as Koorde and DH. Given the constant-degree property, they are well suited for systems with small node degrees. This is due to the fact that a small node degree induces low system maintenance costs. However, when a small diameter is required, the node degree increases.

In our analysis we were interested in the average routing distance and the network size for `Tango` ($k = 3$) and DH with respect to different node degrees. We also plot them for Chord to have a third party reference. To compute the average routing distance for DH we used the μ_d formula for de Bruijn graphs given in [8] and doubled it to achieve load balancing, as suggested in [5]. As shown in Figure 4.2, for the same node degrees (inferior to 34), and almost the same network size, `Tango` provides lower values for the average routing distance than DH. This results are not that surprising since both `Tango` and de Bruijn graphs reduce redundancy in multiplicity of valid paths.

5 Conclusion

First, in this paper we presented a model to better characterize the structure of the current logarithmic-degree P2P exponential structured networks, such as Tapestry, Pastry, Chord and DKS, in terms of absolute and relative exponential structured networks.

On the other hand, we proposed the `Tango` approach to better structure the relative exponential networks to increase their scalability by exploiting the redundancy in the lookup paths. We showed that `Tango` is more scalable than the current logarithmic-based DHTs. We analyzed the structure of `Tango` with respect to the one of DKS and, implicitly, to the one of Chord. Particularly, we observed that, for small values of the exponential factor k, `Tango` is much more scalable than DKS (and Chord), while for big values of k the scalability of the two networks is more comparable. However, since increasing k leads to increasing the resource consuming and the maintenance cost, it is likely that relative small values of k will be employed. We also analyzed `Tango` with respect to DH, a constant-degree network. We observed that, for networks with relative large node degrees, the average routing distance in `Tango` and DH are comparable.

Given its structuring flexibility and its scalability potential, we chose `Tango` to be the algorithm underlying the recently released P2P middleware, called P2PS [11], that we have developed within the framework of our research projects ORAGE and PEPITO. We also implemented several demo applications using P2PS, including a PostIt [12] application and a children's drawing tool called P2P-Matisse [13].

As future work, we plan to address the redundancy in `Tango` resulting from the commutative property of the finger addition operation.

Acknowledgments

We thank Kevin Glynn from Université catholique de Louvain, Belgium and Luc Onana from Swidish Institute of Computer Science, Sweden for their constructive comments.

This work was partially funded at CETIC(http://www.cetic.be) by the Walloon Region (DGTRE) and the European Union (ERDF and ESF), and at UCL by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under IST-2001-33234 PEPITO.

References

[1] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *ACM SIGCOMM*, August 2001.

[2] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In *ICDSP*, November 2001.

[3] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Technical Report CSD-011141, U.C. Berkeley, April 2001.

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

[5] M. Naor and U. Wieder. Novel Architectures for P2P Applications: the Continous-Discrete Approach. In *ACM SPAA*, June 2003.

[6] S. El-Ansary and L. Onana et al. A Framework for Peer-to-Peer Lookup Services based on k-ary Search. Technical Report TR-2002-06, SICS, May 2002.

[7] L. Onana and S. El-Ansary et al. DKS(N, k, f): A Family of Low Communication, Scalable and Fault-Tolerant Infrastructures for P2P Applications. In *CCGRID2003*, May 2003.

[8] D. Loguinov and A. Kumar et al. Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience. In *SIGCOMM*, August 2003.

[9] K. Lakshminarayanan, A. Rao, S. Surana, R. Karp, and I. Stoica. Hyperchord: A Peer-to-Peer Data Location Architecture. Technical Report CS-021208, U.C. Berkeley, December 2001.

[10] V. Mesaros, B. Carton, and P. Van Roy. S-Chord: Using Symmetry to Improve Lookup Efficiency in Chord. In *PDPTA*, June 2003.

[11] P2PS v 1.0, Peer-to-Peer System Library, October 2003. Universtité catholique de Louvain, and CETIC, Belgium. www.mozart-oz.org/mogul/info/cetic_ucl/p2ps.html.

[12] PostIt, May 2003. Universtité catholique de Louvain, and CETIC, Belgium. www.info.ucl.ac.be/-valentin/postit.

[13] P2P-Matisse, June 2003. Universtité catholique de Louvain, and CETIC, Belgium. www.info.ucl.ac.be/people/ned/matisse.