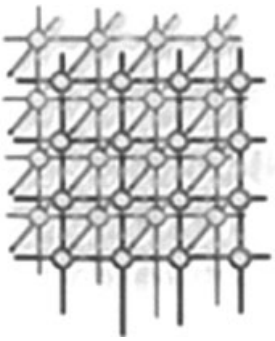
CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2008; **20**:643–657
Published online 17 July 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe.1243



Optimizing the finger tables in Chord-like DHTs

Giovanni Chiola[1,*,†], Gennaro Cordasco[2], Luisa Gargano[2], Alberto Negro[2] and Vittorio Scarano[2]

[1]*DISI, Università di Genova, Italy*
[2]*DIA, Università di Salerno, Italy*

SUMMARY

The Chord protocol is the best known example of implementation of logarithmic complexity routing for structured peer-to-peer networks. Its routing algorithm, however, does not provide an optimal trade-off between resources exploited (the size of the 'finger table') and performance (the average or worst-case number of hops to reach destination). Cordasco *et al.* showed that a finger table based on Fibonacci distances provides lower number of hops with fewer table entries. In this paper we generalize this result, showing how to construct an improved finger table when the objective is to reduce the number of hops, possibly at the expense of an increased size of the finger table. Our results can also be exploited to guarantee low routing time in case a fraction of nodes fails. Copyright © 2007 John Wiley & Sons, Ltd.

Received 9 March 2007; Accepted 7 May 2007

KEY WORDS: peer-to-peer overlay networks; routing efficiency and fault-tolerance; efficient protocol implementation; analytic evaluation; Monte Carlo simulation

1. INTRODUCTION AND BACKGROUND

Chord [1] is the best known, structured, peer-to-peer (P2P) protocol proposed in the literature. It implements the idea of 'consistent hashing' [2] to build a distributed hash table (DHT) based on several independent nodes. In summary, the innovative contribution of Chord was the combination of the 'autonomic features' of a truly distributed P2P protocol with an efficient lookup implemented by a dichotomous search in an ordered overlay.

In particular, the lookup algorithm is based on the construction by each peer of the so-called 'finger table' data structure. Each peer p is uniquely identified by a 160 bit value K_p chosen at random with uniform probability. The value of the key K_p determines the position of the peer p in

*Correspondence to: Giovanni Chiola, DISI, Università di Genova, Italy.
†E-mail: chiolag@acm.org

Contract/grant sponsor: Italian FIRB project Web-MiNDS http://web-minds.consorzio-cini.it/

the virtual ring overlay network so that successors in the ring have higher key values (modulo 2^{160}). The finger table for peer p is an array fing_{K_p} whose ith entry $\text{fing}_{K_p}[i]$ is the network address of the first peer q that can be encountered on the virtual ring overlay such that $K_q \geq (K_p + 2^i) \bmod 2^{160}$. When peer p is asked to route a message toward the target key K_t, it uses the 'greedy' algorithm of forwarding the message to the peer that is included in its finger table at position j such that $\text{fing}_{K_p}[j] \leq K_t$ and $\text{fing}_{K_p}[j+1] > K_t$. Notice that a virtual ring containing a number of peers substantially less than 2^{160} will be characterized by the fact that there exists a positive integer $k < 160$ such that for any peer p and $\forall i < k$, $\text{fing}_{K_p}[i] = \text{fing}_{K_p}[i+1]$. Hence, in practice, the size of the finger table changes as a function of the number of peers in the overlay. In particular, the expected value of the number of peers in the finger table (which we shall call the 'Node Degree') for any peer p is $\log_2(N)$, where N represents the total number of peers in the overlay. The greedy routing applied to such a finger table data structure results in a number of hops that is at most equal to the number of different entries in the tables. This performance measure is referred to as the 'Diameter' of the overlay, and can also be statistically determined by $\log_2(N)$.

Although the general idea behind Chord is very interesting and worth while, the actual implementation of the lookup protocol [1] has some performance and reliability drawbacks. From a theoretical point view, for instance, Cordasco *et al.* [3] proposed a different way of constructing the finger table data structure, which provides a better trade-off between Diameter and Node Degree, thus leading to more efficient routing. From a practical point of view, the algorithm complexity analysis shows that 'strong stabilization' in case of failure may require $O(N^3)$ steps [4], thus preventing the original protocol from properly scaling up in a realistic context where nodes are subject to failure.

In this paper we generalize the idea of Fibonacci distances by defining a family of functions that yield increasing finger tables which are always logarithmic in size, but that have higher constant factors. The idea is that the additional fingers yield increased routing performance in case all nodes are functional. Moreover, even in case a small fraction of nodes fails, thanks to the increased size of the finger table, the routing algorithm will degrade its performance more gracefully as compared to the original one. As shown by our analytical and simulation studies, the increased size of finger tables may improve the reliability of Chord-like rings and reduce the expected number of hops even in case a non-negligible fraction of nodes fails.

The starting point for our work being F-Chord($\frac{1}{2}$) [3], which provides the optimal trade-off between node degree and network diameter, our generalization keeps the minimality of the worst case number of hops, while the average number of hops is reduced by increasing the number of fingers. Monte Carlo simulation results show that our generalization provides superior routing performance compared to two other alternatives already considered in literature, namely, Base-k (see Section 2) and Extended Fibonacci [5].

Subsequently, Cordasco *et al.* [6] extended their neighbor-of-neighbor (NoN) routing algorithm—which is characterized by $O(\log(N)/\log(\log(N)))$ complexity—to F-Chord(α). Indeed the very same extension applies to our generalized finger table as well, but this kind of extension is beyond the scope of this work [7].

The rest of the paper is organized as follows. Section 2 defines in a precise way the 'competition' against which our new results should compare. Section 3 re-derives the idea of F-Chord($\frac{1}{2}$) [3] routing table from a different perspectives, and then generalizes the idea to bases $k > 2$, which represents one of the main contributions of this paper. Section 4 discusses the efficient implementation of

DOI: 10.1002/cpe

the proposed finger table management protocol, providing details useful to interpret the simulation results reported in the following sections. Section 5.1 compares the routing performance estimates computed by Monte Carlo simulation of our generalized tables against Base-k and extended Fibonacci finger tables. Section 5.2 assesses the problem of routing fault tolerance, presenting simulation results in case a fraction up to 10% of the nodes do not respond to routing requests. Finally, Section 6 contains concluding remarks and discusses possible extensions of this work.

2. PRELIMINARIES

The purpose of this section is to define notation and recall some more or less known results that have sporadically appeared in literature, rather than proposing original results.

The finger table in a Chord-like ring can be defined as an array of approximately $\log(N)$ elements (where N is the total number of nodes in the overlay) that contain the network addresses (e.g. IP) of some of the successors of the node along the ring at distances that increase as a function of the index in the table. In particular, if we denote by $J(i)$ the distance (also called jump) of the ith finger from the considered node, in case of Chord we have:

$$J(i) = 2^i$$

For example, if we consider a Chord ring containing $N = 16$ nodes, each one characterized by a unique Id ranging from 0 to 15, each peer will build a finger table containing four elements, defined as for all $n = 0, \ldots, 15$,

$$\text{fing}_n[0] = n + 1 \bmod 16 \quad \text{fing}_n[1] = n + 2 \bmod 16$$
$$\text{fing}_n[2] = n + 4 \bmod 16 \quad \text{fing}_n[3] = n + 8 \bmod 16$$

One way of analyzing the routing complexity in terms of number of hops needed to reach destination from source is to compute the difference (modulo N) between the destination key and the source key, write it in binary form, and compute the number of '1' figures in that representation. In the worst case (e.g. when node 0 is requested to route to node 15), the number of hops required to reach destination is equal to the size of the finger table, i.e. $\log_2(N)$. If we generate routing request at random, with uniform distribution from 0 to $N - 1$, the average number of hops is instead half the size of the finger table, namely $\log_2(N)/2$.

Base-k finger tables: In analogy with the representation Base-k for natural numbers, if one wants to increase the size of the finger table in order to reduce the expected number of routing hops, an obvious approach is to use a base greater than 2. For example, one could construct a finger table whose jumps increase as a power of $k = 3$ rather than as a power of two but that are repeated $k - 1$ times, i.e. for all $l = 0, \ldots, \log_3(N)$,

$$J((k-1)\cdot l) = k^l, \quad J((k-1)\cdot l + 1) = 2 \cdot k^l$$

For example, if we consider a Chord-like ring containing $N = 27$ nodes, each one characterized by a unique Id ranging from 0 to 26, each peer will build a finger table containing 6 ($= (k-1)\cdot\log_3(N)$)

DOI: 10.1002/cpe

elements, defined as for all $n = 0, \ldots, 26$,

$\text{fing}_n[0] = n + 1 \bmod 27$ $\text{fing}_n[1] = n + 2 \bmod 27$
$\text{fing}_n[2] = n + 3 \bmod 27$ $\text{fing}_n[3] = n + 6 \bmod 27$
$\text{fing}_n[4] = n + 9 \bmod 27$ $\text{fing}_n[5] = n + 18 \bmod 27$

Again the easy way of analyzing the hop count is to compute the difference between destination key and source key, represent this difference in Base-k notation, and count the number of figures that are different from '0' in that representation. In our example, one of the worst cases occurs when node 0 has to route a message to node 16. In this case the base-3 representation of the difference between destination and source key is '121', and indeed three hops are required to reach destination following the jumps available in the peers' finger tables. Using the greedy routing, node 0 will forward to node $\text{fing}_0[4] = 9$, then node 9 will forward to node $\text{fing}_9[3] = 15$, and eventually node 15 will forward to $\text{fing}_{15}[0] = 16$.

Definition 2.1. Base-k sequence of jumps.
For any chosen base $k > 2$, for all $l = 0, \ldots, \log_k(N) - 1$, for all $i = 0, \ldots, k - 2$

$$J((k-1) \cdot l + i) = (i+1)k^l$$

By construction, the node degree (i.e. size of the finger table) for Base-k overlays having N peers is $(k-1) \cdot \log_k(N)$. Moreover, an overlay adopting this kind of finger table and adopting a Greedy routing algorithm is characterized by the following properties.

Property 2.1. Diameter (i.e. worst case number of hops) for Base-k overlays with N peers is of the order of $\log_k(N)$.

Proof. Consider two nodes s and t at distance $d(s, t)$. Compute p such that $k^{p-1} < d(s, t) \leq k^p$. Since in the interval $(k^{p-1}, k^p]$ there are $k - 1$ jumps at distance k^{p-1}, after one (Greedy) jump the distance reduces to less than k^{p-1}. By iterating, we have that after i jumps the distance reduces to less than k^{p-i}. Since $d(s, t) < N$, we have that $p \leq \lceil \log_k(N) \rceil$, then after $\lceil \log_k(N) \rceil$ jumps the distance reduces to less than $k^{p - \lceil \log_k(N) \rceil} \leq 1$. □

Property 2.2. Average number of hops (for uniformly distributed random routing requests) for Base-k overlays with N peers is of the order of $(k-1)/k \cdot \log_k(N)$.

Proof. Assume $N = k^p$ and consider a generic lookup request. Partition the routing in phases as follows: in phase i, the distance between the current source and the final target belongs to $(k^{i-1}, k^i]$. From the proof of Property 2.1 we know that we need at most one jump to move from phase $i + 1$ to phase i. Moreover, after a jump from phase $i + 1$, if the remaining distance is less than k^{i-1}, then we reach a phase $j < i$ without any jump in phase i. In particular, the probability that a jump of size k^{i-1} is needed is $(k^i - k^{i-1})/k^i = (k-1)/k$. □

For small values of k, e.g. $k = 3$ or $k = 4$, this extension of the finger table may provide substantial performance improvements compared to the standard Base-2 tables (i.e. Chord), while increasing the size of the finger table by a relatively small multiplicative factor (1.5 in case $k = 4$). For larger values of k, instead, a diminishing return effect is expected, with almost linear increases of the

DOI: 10.1002/cpe

finger table size (of the order of $k/\log(k)$) only partially compensated by very modest reductions of the path length (of the order of $1/\log(k)$).

Finger tables based on Fibonacci jumps: one of the distinctive characteristics of a finger table based on the Fibonacci sequence as the definition of the successive jumps $J(i)$ is that the difference between two consecutive jumps is the preceding jump, i.e. $\forall i \in \mathbb{N}$

$$J(i+2) - J(i+1) = J(i)$$

This relation has the following interesting consequence, when the finger table is adopted to support a Greedy routing algorithm.

Property 2.3. Diameter with homogeneous Fibonacci jumps. If a routing request for node $v + x$ is issued to node x, and if $J(l+1) < v < J(l+2)$, then in one hop this request is forwarded to node $y = x + J(l+1)$, and on the latter node the following inequality holds true: $w < J(l)$, where $w = v - J(l+1)$. As a consequence, the maximum number of hops will never exceed half the size of the finger table.

This observation yields to the idea that half of the fingers can be removed from the table (either the ones with odd index or the ones with even index) without increasing the diameter of the overlay network. Taking only the odd (or even) numbers of a Fibonacci sequence as the jumps to define the finger table yields what in [3] was called F-Chord(α) in the particular case of $\alpha = \frac{1}{2}$. This was proved to be the optimal trade-off between node degree (size of the finger table) and diameter (the worst case number of hops adopting the Greedy algorithm). Indeed in this case if a routing request for node $x + v$ is issued to node x, and if $J(l) < v < J(l+1)$, then in two hops this request is forwarded to node $y = x + J(l) + J(m)$ where $m \leq l$. In either case, on the latter node, the following inequality holds true: $w < J(l-1)$, where $w = v - J(l) - J(m)$.

Hence, the worst case number of hops is again equal to the size of finger table as in Base-2, i.e. Chord. However, in this case the size of the finger table is 28% smaller than Chord's. Therefore, F-Chord($\frac{1}{2}$) asymptotically achieves a diameter 28% lower than Chord using a finger table 28% smaller than Chord. Notice that this property is not enjoyed by Base-k jumps.

Extended Fibonacci distances: A first attempt to extend Fibonacci sequence of jumps in order to reduce the average number of hops at the expense of larger finger tables was presented in [5]. The idea was simply to reduce the speed of the increase of the jumps by using a recursive definition of the type

$$J(i+1) = J(i) + J(i-k)$$

with $k \geq 1$. This sequence, which we will henceforth call F-k, was empirically studied by simulation and showed to be able to reduce the average path length for increasing values of k. Unfortunately, the increase in size of the finger table prevented the use of large values for k.

3. FINGER TABLES BASED ON GENERALIZED F-CHORD($\frac{1}{2}$) JUMPS

In this section, we present a completely different approach to the extension of Fibonacci-type sequences of jumps. In particular, we generalize the property of F-Chord($\frac{1}{2}$) to bases $k > 2$.

DOI: 10.1002/cpe

In the following we show by construction that the sequence of jumps is such that the minimal diameter is produced over the maximum range of ring size.

Definition 3.1. Range of the ring with guaranteed diameter h is denoted as $R(h)$. $R(h)$ is the maximum value $v \in \mathbb{N}$ such that $\forall w \in [0, v)$ a routing request for $x + w$ can be routed to destination in no more than h hops (i.e. $R(h)$ is the size of the greatest ring with diameter h).

For the sake of simplicity, let us consider first the case of a routing table derived by Base-3 jumps, as defined in 2.1. A random request can be routed in zero hops only if the request is for the node itself, hence $R(0) = 1$. On the other hand, a routing request can be routed in no more than one hop only if the request is for the node itself or for one of the nodes that Id is one of the fingers. If we consider the sequence of jumps for Base-3 finger tables, we can easily realize that $R(1) = 4 = k + 1$. This is simply because there is no jump in the finger table corresponding to the value $k + 1$.

Consider now the cases in which the request can be routed to destination in no more than 2 hops. We may observe that the two jumps following $k = 3$ are such that $J(3) - J(2) = J(4) - J(3) = 3 = R(1) - 1$. This is not optimal from the point of view of maximizing $R(2)$, as if for instance we have to route a value v such that $J(2) < v < J(3)$ we would forward the request to node $x + J(2)$ with a remaining displacement $w = v - J(2) < R(1) - 1$. Indeed, if we defined $J(3) = 7$ and $J(4) = 11$ (instead of 6 and 9), we would have obtained $R(2) = 15$ instead of 13. The improvement seems marginal in case of $h = 2$ (where the range is increased only by two compared to Base-3). However, the difference becomes substantial when we consider higher values of h. In general, the jumps for the case $k \geq 2$ are recursively defined as follows.

Definition 3.2. MaxRange Base-k (M-k) (i.e. generalized Base-k yielding minimal diameter and maximum range):

$$
\begin{aligned}
&R(0) = 1, \quad J(0) = 1 \text{ for each integer } l \geq 0\\
&\quad \text{for each } i = 1, \ldots, k-1,\\
&\qquad J((k-1)\cdot l + i) = J((k-1)\cdot l) + i \cdot R(l)\\
&\quad R(l+1) = J((k-1)\cdot l) + k \cdot R(l)
\end{aligned}
$$

Notice that in the particular case $k = 2$, the sequence of jumps defined above is: 1, 2, 5, 13, 44, ..., which is the Fibonacci sequence without odd index numbers. Hence, MaxRange Base-k is the proper generalization of F-Chord($\frac{1}{2}$) to a base $k > 2$.

For example, a MaxRange Base-3 finger table for peers belonging to a system containing $N = 56$ peers would be defined as follows:
for all $n = 0, \ldots, 55$, $\text{fing}_n[0] = n + 1 \bmod 56$

$\text{fing}_n[1] = n + 2 \bmod 56$ $\text{fing}_n[2] = n + 3 \bmod 56$ $(R(0) = 1)$
$\text{fing}_n[3] = n + 7 \bmod 56$ $\text{fing}_n[4] = n + 11 \bmod 56$ $(R(1) = 4)$
$\text{fing}_n[5] = n + 26 \bmod 56$ $\text{fing}_n[6] = n + 41 \bmod 56$ $(R(2) = 15)$

The reader is urged to verify that the routing based on this finger table requires at most three hops, exactly as in the case of a Base-3 system containing only 27 peers.

DOI: 10.1002/cpe

Now let us derive some interesting properties of the range function.

Property 3.1. For each $l \geq 1$, $R(l+1) = (k+1) \cdot R(l) - R(l-1)$.

Proof. By substitution of $J((k-1) \cdot l) = J((k-1) \cdot (l-1) + (k-1)) = J((k-1)(l-1)) + (k-1) \cdot R(l-1) = R(l) - R(l-1)$ in Definition 3.2. □

Property 3.2. For each $l \geq 0$,

$$R(l) = \frac{\alpha^{l+1} - \beta^{l+1}}{\alpha - \beta}$$

where

$$\alpha = \frac{(k+1) + \sqrt{(k+1)^2 - 4}}{2} \quad \text{and} \quad \beta = \frac{(k+1) - \sqrt{(k+1)^2 - 4}}{2}$$

Proof. The property is the general solution of the linear recurrence relation of order 2, with initial values, respectively, 1 and $k+1$, where α and β are the two roots of the characteristic equation $x^2 - (k+1) \cdot x + 1$. □

Property 3.3. Let d be the diameter of Base-k system with $N = R(d)$ nodes,

$$\log_{k+k/(k+1)} \left(\left(\frac{k^2 - 1}{k^2} \right) \cdot N \right) < d < \log_{k+(k-1)/k} (N+1)$$

Proof. By (1) and observing that

$$\left(k + \frac{k-1}{k} \right) < \alpha < \left(k + \frac{k}{k+1} \right) \quad \text{and} \quad \frac{1}{k+1} < \beta < \frac{1}{k}$$

we have

$$\left(k + \frac{k-1}{k} \right)^d - 1 < N < \left(\frac{k^2}{k^2 - 1} \right) \cdot \left(k + \frac{k}{k+1} \right)^d$$ □

Property 3.4. Let δ be the node degree of Base-k with $N = R(d)$ nodes,

$$(k-1) \cdot \log_{k+k/(k+1)} \left(\left(\frac{k^2 - 1}{k^2} \right) \cdot N \right) + 1 < \delta < (k-1) \cdot \log_{k+(k-1)/k} (N+1) + 1$$

Proof. By Definition 3.2 we have $\delta = (k-1)d + 1$. □

Property 3.5. Let APL be the average path length (i.e. number of hops for uniformly distributed random routing requests) of a MaxRange Base-k system with $N = R(d)$ nodes,

$$\left(\frac{k-1}{k} \right) \cdot \log_{k+k/(k+1)} \left(\left(\frac{k^2 - 1}{k^2} \right) \cdot N \right) < \text{APL} < \left(\frac{k+2}{k+3} \right) \cdot \log_{k+(k-1)/k} (N+1)$$

DOI: 10.1002/cpe

Proof. Consider a generic lookup request, and partition the routing in phases as follows: in phase i the distance between current source and final target is less than $R(i)$. If in phase i the distance is less than or equal to $J((k-1)(i-1))$, then we can move from phase i to phase $i-1$ without any jump. Therefore, at each phase i a jump is actually required to move to phase $i-1$ with probability at most

$$\frac{R(i) - J((k-1)(i-1))}{R(i)} = \frac{R(i) - R(i-1) + R(i-2)}{R(i)}$$

Hence, we have

$$\text{APL} < \sum_{i=1}^{d} \frac{R(i) - R(i-1) + R(i-2)}{R(i)} < \left(\frac{k+2}{k+3}\right) \cdot d < \left(\frac{k+2}{k+3}\right) \cdot \log_{k+(k-1)/k}(N+1)$$

Because of the UB in Property 3.3. By using the same argument and the corresponding lower bound in Property 3.3 we can easily show the lower bound on APL. □

In order to quantify the improvement in range introduced by this definition of jumps, let us consider the curves depicted in Figure 1. They show the growth of the size of the finger table as a

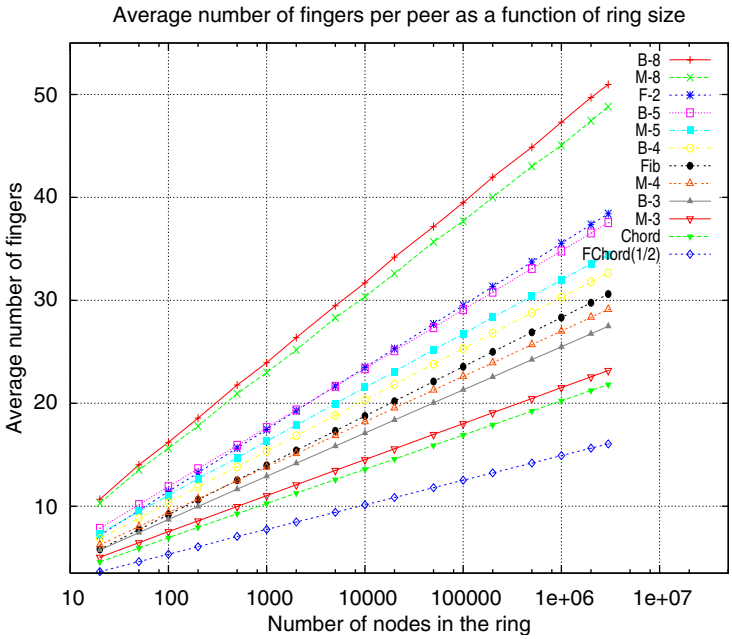


Figure 1. Number of elements in the finger table as a function of the number of nodes in the ring, where (as in the following figures) B-k denotes Base-k, M-k denotes generalized MaxRange minimal diameter Base-k, Fib stands for Fibonacci, and F-k denotes 'extended Fibonacci.'

DOI: 10.1002/cpe

function of the increase in the number of nodes in the overlay, simulated over the range from $N = 20$ up to $N = 3\,000\,000$. Notice that the curve corresponding to the generalized MaxRange minimal diameter Base-k (M-k) is not only lower than the one corresponding to Base-k (B-k), but also quite close to the one corresponding to Base-$(k-1)$, while the diameter is guaranteed by definition (see Definition 3.1) to be lower than the one of Base-k as well.

In Figure 1, the size of the finger table for the 'extended Fibonacci' functions as proposed in [5] is also reported for the sake of comparison. A comparison in terms of average number of hops is presented in Section 5.1 based on Monte Carlo simulation.

4. FINGER TABLE MANAGEMENT IN CHORD-LIKE PROTOCOLS

As already pointed out in [4], the finger table in a Chord-like routing protocol does not necessarily have to be up to date in order to guarantee the correctness of the procedure. The only correctness requirement is the link to the next peer in the ring, namely the first entry in the finger table. An 'error' in the first entry of the finger table may lead to the inability to properly route a message to destination, while an error in the subsequent entries of the finger table may simply slow down the routing algorithm by unnecessarily increasing the number of hops needed to complete the route. Moreover, a 'statistical' form of adjustment of the distances of fingers may be sufficient to guarantee good performance.

Therefore, our suggestion for managing the finger table is to only keep the first entry in the finger table constantly up to date by a proper synchronization protocol between any pairs of subsequent peers in the ring. As soon as an anomaly is discovered in this link (namely during the addition of a new node or the removal of one node from the ring), that portion of the ring may be temporarily marked as 'under repair.' Any route request coming to the node that was responsible for that subset of keys may be delayed until the link is updated if the node is 'under repair.' By using this approach, one could remove the (global) 'stabilization procedure' that was originally adopted in Chord. Such a stabilization procedure could affect availability as well as performance in case a relatively large fraction of nodes attach and/or detach themselves to/from the ring.

All subsequent entries in the finger table may be updated periodically with a periodicity that is selected as a functional parameter of the system. The addition of a few nodes to the ring will change the actual distance of the peers that are linked in the finger table with respect to the 'ideal' distance requested by the jump definition $J(i)$. However, such a temporary divergence will only (marginally) affect performance, and only until the entries of the finger table are recomputed. Simulation results show that the effect of the 'precision of distance function' in terms of routing performance decreases as the index increases, so that updating the finger table could be carried out incrementally, with the nearest fingers updated more frequently and the furthest fingers only seldom updated.

On the other hand, the removal of some peers from the ring could leave in the finger table of some other peers the address of nodes that have left, and that cannot respond to queries any more. In order to deal with this inconvenience, a 'fault-tolerant' version of the Greedy routing can be adopted, which chooses the 'next closer finger' in the table if the optimal one does not respond within a predefined time-out. Since the first finger is guaranteed to be alive except for the short time when the link is under repair, the fault-tolerant greedy routing allows one to use an out-of-date

DOI: 10.1002/cpe

finger table with degraded performance. Simulation results to assess the amount of performance degradation are also presented in Section 5.2.

The choice of performing periodic updates of the finger table and allowing their entries to be slightly out of date has a beneficial effect on the requirements placed on the size of the finger table. In case all elements have to be kept up to date, the size of the finger table must be kept to a minimum in order to reduce the overhead associated with join/leave operations. On the other hand, if a divergence from the optimal is tolerated for some time, then the size of the finger table may grow without major impact on the ring maintenance protocol overhead. The only practical constraint on the growth of the finger table is that scalability is maintained (by keeping the growth logarithmic in the total number of peers).

A final implementation problem worth mentioning is related to the sparsity of the *Id*s associated with the peers compared to the *Id* space. Normally the space of node *Id*s is huge compared to the actual number of peers in the ring, so that the address space is almost empty. This fact associated with the use of cryptographic Hash functions guarantees the absence of collisions in the *Id* space. However, this property may give rise to various implementations of the finger table starting from the same theoretical definition of jump sequences.

In our simulation code we adopted the following choice to map the sequence of jumps into the actual finger table of each node. The definition of the jumps is mapped to the node *Id* space, and the mapped jump values are normalized by a multiplicative coefficient so that the range of the last normalized jump $R(h_{max})$ is equal to the highest *Id* in the *Id* space. A lookup operation is simulated to identify the peer that is responsible for the given normalized jump value, and the address of this peer is adopted as the ith entry in the finger table.

The filling of the finger table starts with the highest index jump, and continues until the immediate successor of the current node is inserted in the table. The size of the finger table is thus automatically adjusted to various sizes of the ring. Due to the random assignment of *Id*s to peers, the finger table of one particular node has not necessarily the same size of the table of another node in the same ring. The actual number of fingers for a particular node depends on the actual distribution of *Id*s of its neighbors. For this reason, the size of the finger table may only be defined in a statistical way, by computing the average over all peers of the actual size of their individual finger tables. This explains why we referred to the size of the finger table as the 'average number of fingers per peer' in Figure 1.

5. MONTE CARLO SIMULATION RESULTS

In order to assess the effectiveness of our proposed finger tables, we used a very simple Monte Carlo simulation approach. A virtual ring is constructed by randomly generating the *Id*s associated with the prescribed number of nodes. Then the finger list for each node is constructed according to the chosen distance function. Finally, a large number of random, uniformly distributed, independent routing requests are issued to the node with the lowest *Id* in the ring, and routed through the nodes simulating the Greedy routing protocol. The number of hops is counted for each route, and statistics are collected. Ninety-nine percent confidence level intervals are estimated in order to ensure the precision of the simulated results. The experiments are repeated until the confidence

DOI: 10.1002/cpe

intervals become small enough. All the results reported in the paper have three digits precision with 99% confidence level.

We first evaluate the performance of the greedy routing algorithm in the ideal case where all peers are functional. Subsequently, we study the performance degradation that occurs in case some of the peers disconnect from the ring and the finger tables are not promptly updated. In this case some of the fingers contacted by function peers will appear as faulty, thus implying the elapsing of communication time-outs and the choice of sub-optimal paths.

5.1. Routing without failures

The results obtained by our simulation experiments in the case all peers are functional are depicted in Figure 2. Both average values and 95 percentile are depicted, the latter being defined as the least number of hops that is not exceeded in at least 95% of the routing requests. As one can see from the diagrams, M-2 (which is F-Chord($\frac{1}{2}$)) yields worse performance than B-2 (which is the original Chord), due to its substantial reduction in the size of the finger table. Notice, however, that the disadvantage of M-2 compared to B-2 appears not to increase in absolute value for large ring sizes.

As we increase the base to values $k>2$, we notice that the performance gap of M-k with respect to B-k vanishes, with M-3 already quite close to B-3 for smaller rings and practically superimposed for larger rings. Moreover, starting from $k=4$, a cross-over point can be identified from which M-k starts outperforming B-k for larger rings. Actually, the cross-over point is already visible in case of the average number of hops for M-4 at 2 000 000 peers, which outperforms B-4 of about 3% in case of 3 000 000 peers. Comparing M-5 against B-5, we can see that B-5 starts with a very minor advantage over M-5 for very small rings, then the two practically overlap for rings containing about 50 000 peers, while M-5 clearly overcomes B-5 in terms of average number of hops when the ring exceeds 500 000 peers, despite the fact M-5 tends to have a degree closer to the one of B-4 than to the one of B-5.

The comparison of MaxRange against (extended) Fibonacci shows that M-4 is superior to Fibonacci (F-1) and M-5 is superior to F-2, in spite of the quite lower number of fingers.

5.2. Rings with failures

One possible drawback of our proposed finger table management algorithm with lazy update is the possibility that one or more links point to peers that disconnected from the ring after the last finger table update. In order to deal with this case, the routing algorithm must be prepared to interact with peers that are not ready to communicate. Notice, however, that the first finger (i.e. the immediate successor) is assumed to be constantly updated, so that each searched key is always associated with a functioning node, and no lookup will ever fail: the only consequence of a non-up-to-date finger table is a possible slow down of the lookup itself, with higher number of hops as compared to the 'normal case' studied in the previous section.

Our proposed solution is to set up a time-out after forwarding a routing request to a peer and reset it upon receipt of acknowledgment. If no acknowledgment is received before the time-out elapses, then the contacted peer is assumed not to belong to the ring, and the routing request is forwarded to the peer whose *Id* is immediately before the failed one in the finger table (i.e. the one

DOI: 10.1002/cpe

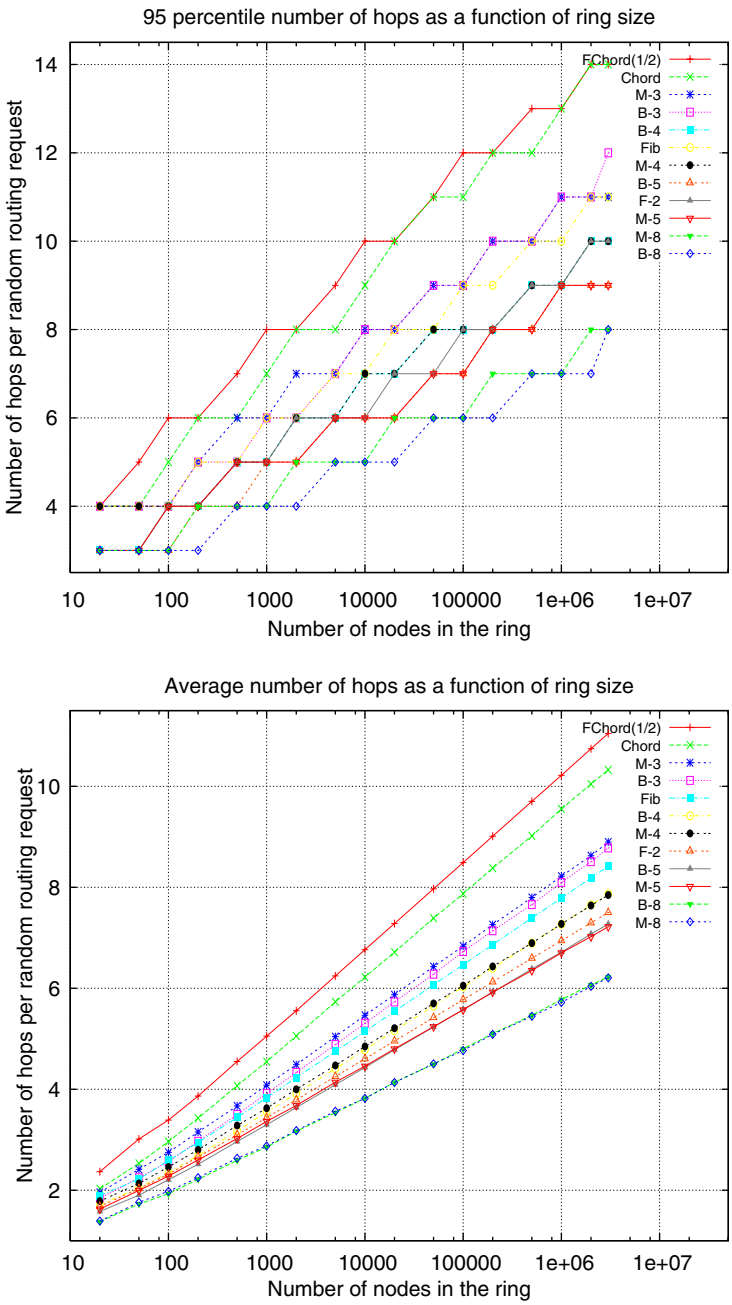


Figure 2. Average and 95 percentile number of hops for random routing requests as a function of the number of nodes in the ring, for the considered finger tables.

DOI: 10.1002/cpe

at a distance immediately shorter than the 'optimal' one that did not respond). In case of multiple failures, the node will keep trying closer and closer neighbors, possibly until the first element in the finger table is reached (remember that this link is assumed to be kept constantly up to date, so that progress of at least one step in the ring is always guaranteed, unless this portion of the ring is momentarily under repair).

In our simulations we assumed that the time-out is set to only three times the average hop time (i.e. 1.5 times the average round-trip). This time-out is added each time the simulated routing algorithm makes an attempt to forward a request to a non-operational node before sending the message to the next suboptimal finger. In practice, larger time-outs should be used to accommodate for the variance that is usually found in round-trip times, so that our results can be interpreted as 'lower bounds' on the expected effect of time-outs.

The expected number of hops obtained by our simulation experiments are depicted in Figure 3, for varying fractions of failed nodes ranging from 0 to 35% in case of rings with 10 000 peers. Figure 4 reports the expected routing time under the assumption that each time-out adds a delay, which is three times the average hop delay. The most interesting observation that one can make is that the M-k family of finger tables is less sensitive to failures than both B-k and F-k. This is probably due to the more efficient use of the fingers that is made, which on one hand reduces the number of hops (hence reducing the probability of getting a failed node in the route), and on the other hand reduces the size of the finger table (thus reducing the total number of failed fingers in case of relatively small failure rates).

More precisely, we conjecture that for very small fraction of failed nodes, the probability of failure of at least one finger is substantially smaller in the M-k family of finger tables due to their smaller size. For larger fractions of failed nodes, the probability of encountering failed fingers in the table becomes comparable, but at this point the lower value of 95 percentile number of hops for the M-k family reduces the worst case probability of incurring in large number of time-outs. As the probability of time-out increases exponentially with the number of hops, even a minor improvement in the distribution of the number of hops may yield substantial advantage in terms of reduction of the expected routing time.

6. CONCLUSIONS

We extended the F-Chord($\frac{1}{2}$) finger table to base $k \geq 2$. We studied the characteristics of the family of functions both analytically and by Monte Carlo simulation. The parameter k may be increased in order to reduce average and maximum number of hops, at the expense of an increased size of the finger table for a given number of peers. We showed that even starting from relatively small values of k the expected number of hops is reduced compared to other finger table structures for large rings, even if the number of fingers is also reduced.

We studied the effect of the combination of our proposed (optimal) finger tables in conjunction with a fault-tolerant version of the greedy routing and lazy update of the tables. The result was also very positive. The same efficiency in the finger table that improves the performance in ideal case, also makes our finger table structure less sensitive to finger failures, thus allowing a larger 'laziness margin' in the update without major performance degradation.

DOI: 10.1002/cpe

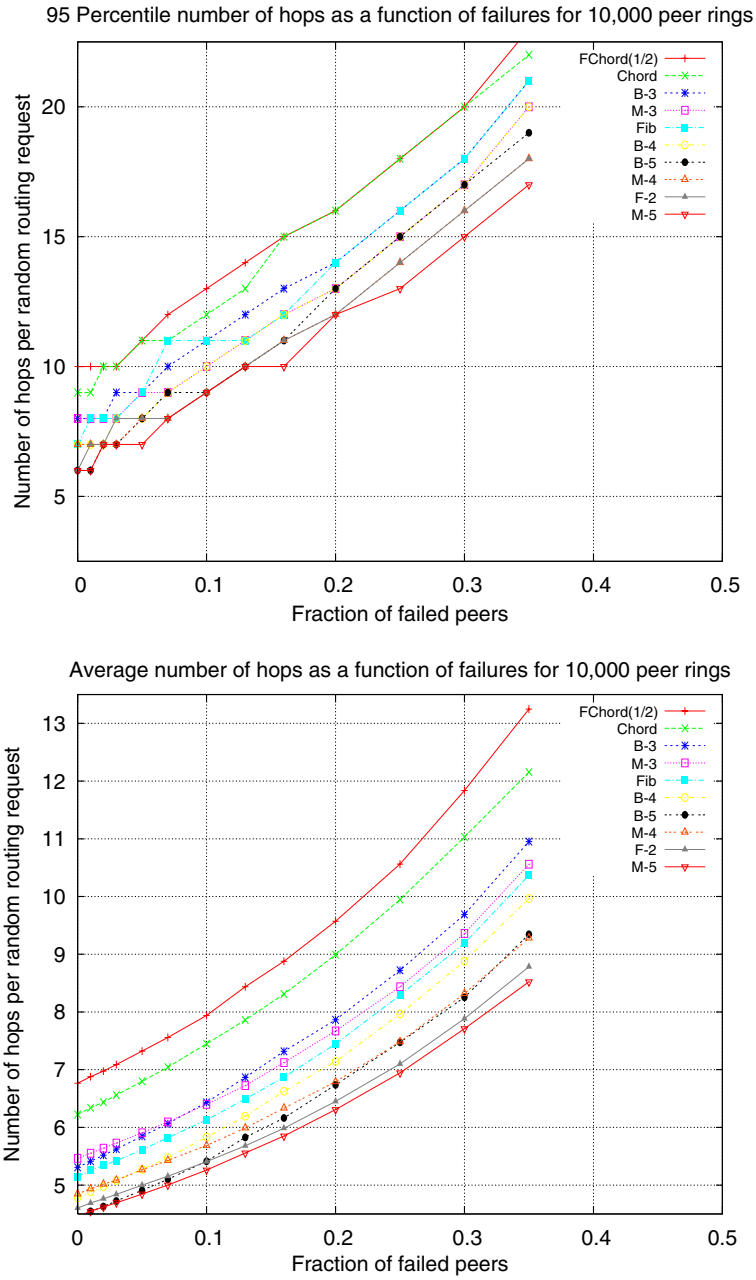


Figure 3. Average and 95 percentile number of hops for random routing requests in a 10 000 node ring as a function of the fraction of nodes that failed.

DOI: 10.1002/cpe

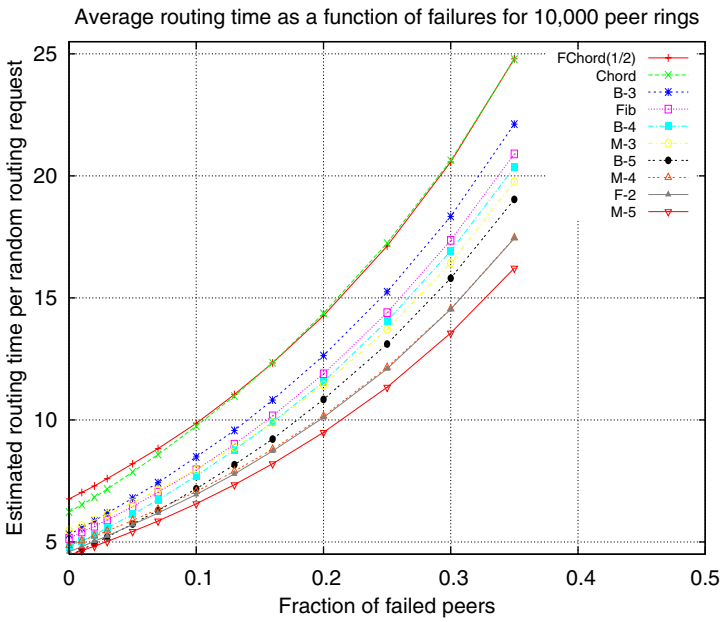


Figure 4. Estimated routing time per routing request (hop time = 1, time-out = 3) in a 10 000 peer ring as a function of the fraction of nodes that failed.

An interesting development of this work is the application of our proposed jump functions to pseudo-randomized finger tables associated with a neighbor-of-neighbor (NoN) routing algorithm, as already done in [6] for the case of F-Chord(α).

REFERENCES

1. Stoica I, Morris R, Liben-Nowell D, Karger DR, Kaashoek MF, Dabek F, Balakrishnan H. Chord: A scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking* (*TON*) 2003; **11**(1):17–32.
2. Karger DR, Lehman E, Leighton FT, Panigrahy R, Levine MS, Lewin D. Consistent hashing sand random trees: Distributed caching protocols for relieving hot spots on the world wide web. *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*. ACM Press: El Paso, TX, U.S.A., 1997; 654–663.
3. Cordasco G, Gargano L, Hammar M, Negro A, Scarano V. F-Chord: Improved uniform routing on chord. *Proceedings of 11th Colloquium on Structural Information and Communication Complexity* (*Sirocco '04*), Smolenice, Slovakia. Springer: Berlin, June 2004; 89–98.
4. Stoica I, Morris R, Karger DR, Balakrishnan H. Chord: A scalable peer-to-peer lookup protocol for internet applications. *Technical Report TR-819*, MIT, LCS, April 2001.
5. Chiola G. Extended Fibonacci distances for fault-tolerant routing in chord-like DHTs. *Proceedings of 1st International Workshop on Hot Topics in Peer-to-Peer Systems* (*HOT-P2P '04*) (in conjunction with MASCOTS 2004), Volendam, The Netherlands. IEEE Computer Society: Silver Spring, MD, October 2004; 10–15.
6. Cordasco G, Gargano L, Hammar M, Negro A, Scarano V. Non-uniform deterministic routing on F-Chord(α). *Proceedings of 1st International Workshop on Hot Topics in Peer-to-Peer Systems* (*HOT-P2P '04*) (in conjunction with MASCOTS 2004), Volendam, The Netherlands. IEEE Computer Society: Silver Spring, MD, October 2004; 16–21.
7. Chiola G, Cordasco G, Gargano L, Hammar M, Negro A, Scarano V. Degree-optimal routing for P2P systems. *Theory of Computing Systems* (*TOCS*), accepted.

DOI: 10.1002/cpe