F-Chord: Improved Uniform Routing on Chord (Extended Abstract)

G. Cordasco⋆ , L. Gargano∗, M. Hammar∗, A. Negro∗, V. Scarano∗

Dipartimento di Informatica ed Applicazioni
Università di Salerno,
84081, Baronissi (SA) – Italy
E-mail: {cordasco,lg,hammar,alberto,vitsca}@dia.unisa.it

Abstract. We propose a family of novel schemes based on Chord retaining all positive aspects that made Chord a popular topology for routing in P2P networks. The schemes, based on the Fibonacci number system, allow to improve on the maximum/average number of hops for lookups and the routing table size per node.

1 Introduction

In this paper, we propose a family of new routing schemes that reduce the routing table size, and the maximum/average number of hops for lookup requests in Chord–like systems [15] without introducing any other protocol overhead. The improvement is obtained with no harm to the simplicity and ease of programming that are some of the many good characteristics that made Chord a popular choice.

The basis of Chord can be seen as a ring of N identifiers labelled from 0 to $N-1$. The edges, representing the overlay network, go from identifier x to identifier[1] $x + 2^i$, for each $x \in \{0, \dots, N-1\}$ and $i < \log N$. The degree and the diameter are $\log N$, the average path length is $(\log N)/2$. Routing is greedy, never overshooting the destination.

Because of low diameter and average path length, Chord offers fast lookup algorithms. By having also low degree, it provides efficient join/leave of nodes since the cost depends on the diameter and the degree and is, in fact, upper bounded by their product. Chord is scalable: with $n \leq N$ nodes present in the network the same performance (in terms of n rather than N) can be obtained

⋆ Work partially supported by EU RTN project ARACNE and by Italian FIRB project WebMinds.

[1] Throughout the paper, arithmetics on node identities is always mod N where N is the number of identifiers. Similarly, all the logarithms are base 2, unless differently specified.

w.h.p.. Efficient routing in Chord is easy (a greedy algorithm is optimal) due to the fact that Chord is *uniform*: x is connected to y iff $x + z$ is connected to $y + z$. Since we are restricting ourselves to uniform routing schemes, we can use the term *jump of size s* to indicate the existence in the overlay network of an edge from x to $x + s$ for any identifier $x = 0, \ldots, N - 1$ (e.g. Chord has jumps 2^i, for $i < \log N$).

Uniformity is a crucial requirement, since it makes any system a good candidate for real implementations: besides simplicity in the implementation it also offers an optimal greedy routing algorithm without node congestion [16]. On the other hand, it is known that, as long as uniformity is required, the $O(\log N)$ values for the number of jumps and diameter cannot be asymptotically improved. Chord's values of the three main parameters (degree vs. diameter and average path length) can be improved if one removes the uniformity request [4, 7–10, 12].

Because of its practicality and its known bounds, it assumes a certain relevance to improve the performance of Chord while retaining simplicity and scalability. Moreover, due to the above, it is interesting and practically useful to improve known results for uniform systems, even only by constant factors. Our objective is to show improved bounds on all the important parameters of a P2P system that affect lookup time and join/leave cost, i.e., degree, diameter and average path length. To this aim we will propose and analyze a novel family of uniform routing schemes.

The lookup process in Chord can be seen as a binary search on an interval of N elements. A natural question to ask is whether the lookup can be realized with a more efficient search technique that can be translated into a uniform overlay network. Efficiency is measured in terms of degree and maximum/average path length. We notice that the problem poses several restrictions on the search model since we are assuming that all nodes are alike and, therefore, only queries taken from a globally given set can be used at any step.

In this paper, we consider search techniques in the above model imposed by uniform routing algorithms (that is, when a fixed set of jumps is available). To the best of our knowledge, while the problem is related to several search problems investigated in the literature, no useful results on problems totally fitting the above model and goals are known (see [3] and references therein quoted). A previous work in the P2P context in this direction is contained in [16]. Our starting point is Fibonacci search [2].

1.1 Our results:

Let $Fib(i)$ be the i-th Fibonacci number. We prove that any uniform algorithm that uses up to δ jumps and has diameter d can reach at most $N(\delta, d)$ consecutive identifiers where $N(\delta, d) \leq Fib(\delta + d + 1)$. This gives us a tradeoff of $1.44 \log N$ on the sum of the degree and the diameter in any P2P network using uniform routing on N identifiers. We then show a family of routing scheme F-Chord(α) that, besides improving all Chord parameters, also reaches equality for any choice of the parameters in the inequality above with $|\delta - d| \leq 1$.

Our analysis has been carried out by considering the N-size identifier space. We use a standard technique (see [15] for technical details) to manage the situation when $n < N$ nodes are in the network. In this case, in fact, we assume that the n nodes are uniformly distributed at random on the N identifiers, therefore we can consider a ring of n "chunks" each with N/n identifiers.

Since in each chunk there are at most $O(\log n)$ nodes w.h.p., any deterministic result on the diameter in terms of N can be easily translated in the same result in terms of n, with high probability. The same argument can be used for the degree. In fact, the distance between two nodes is at least N/n^2 w.h.p.. If $\delta(N)$ is the degree when all the N identifiers are used, then the number of finger pointers that are useful (and consequently stored) with n nodes is at most $2\delta(N)$, w.h.p..

1.2 Related work

The Chord system was introduced in [15] to allow efficient lookup in a Distributed Hash Table (DHT). By using logarithmic size routing tables in each node, Chord allows to find in logarithmic n number of routing hops the node of a P2P system that is responsible for a given key. Adding or removing a node is accomplished at a cost of $O(\log^2 N)$ messages. A thorough study of uniform system is given in [16].

In general, Chord can be improved at the expenses of uniformity [7, 9, 10]. Recently, some non-greedy routing algorithms were proposed, that use De-Bruijn based DHT [4, 11], whose goal is to reach an optimal trade-off between degree and path length and, in particular, allow routing in $\Omega(\log N/ \log\log N)$ with logarithmic degree.

One can also improve the results by eliminating the deterministic requirement. In fact, it is possible (see [8, 12]) to route greedily in $\Theta(\log N/ \log\log N)$ with logarithmic degree by using randomization and the so called *neighbor-of-neighbor (NoN) approach*: a node uses, at each step, its neighbor's neighbors to make greedy decisions. It is not difficult to see that these techniques can be easily adapted to our scheme thus obtaining similar results.

However, in this paper we focus on *deterministic* and *uniform* routing schemes. Among other advantages, they offer an optimal greedy routing strategy that provides simplicity, fault tolerance (as long as some node has edges toward destination, the routing succeeds) and locality (messages flow only on the portion of ring between source and destination), as noted in [12].

1.3 Organization of the paper

The rest of the paper is organized as follows. First of all, we provide the proof of the lower bound in Section 3. Then, in Section 4 we introduce the F-Chord(α) family and in Section 5 we prove some of its properties with regards to the degree, diameter, average number of hops, and congestion. Finally, in Section 6 we conclude with some remarks and give some open problems.

2 Fibonacci numbers

We, briefly, recall here some basic facts on Fibonacci numbers which will be used in the sequel (see [6]). Let $Fib(i)$ denote the i-th Fibonacci number. They are defined as $Fib(0) = 0, Fib(1) = 1$ and, for each $i>1$,

$$Fib(i) = Fib(i-1) + Fib(i-2).$$

For each index i, it holds

$$Fib(i) = [\phi^i/\sqrt{5}],$$

where $\phi = 1.618\ldots$ is the golden ratio and [ ] represents the nearest integer function. Furthermore,

$$\sum_{i=0}^{p} Fib(i) \cdot Fib(p-i) = \frac{1}{5}\{p[Fib(p+1) + Fib(p-1)] - Fib(p)\}.$$

3 The lower bound

In this section, we furnish a tradeoff of $1.44 \log N$ on the sum of the degree and the diameter in any P2P network using uniform routing on N identifiers. Namely, we prove the following theorem.

Theorem 1 *Let $N(\delta, d)$ denote the maximum number of consecutive identifiers obtainable trough a uniform algorithm using up to δ jumps (i.e. degree δ) and diameter d. For any $\delta \geq 0$, and $d \geq 0$, it holds that*

$$N(\delta, d) \leq Fib(\delta + d + 1) \quad (1)$$

> **Remark:** *It can be shown that inequality (1) is strict whenever $|\delta - d| > 1$. On the other hand, we will exhibit a routing scheme that reaches equality for any choice of the parameters with $|\delta - d| \leq 1$.*

Proof. We proceed by induction on the sum $\delta + d$. Trivially, $N(0,0) = N(1,0) = N(0,1) = 1 = Fib(1) = Fib(2)$. Assume that (1) holds for any δ and d with $\delta + d < x$. We will show that for any degree y and diameter z with $y + z = x$ it holds that $N(y, z) \leq Fib(x + 1)$. We distinguish three cases on the number of times the first (i.e. the biggest) jump is repeated. We recall that the assumption of a uniform algorithm implies that jumps can be used to build paths in a greedy manner with respect to their size, that is, on each path jumps of the same size can be assumed to be consecutive.

Case 1: the first jump appears at most once on each path. In this case, each path, either starting with the given jump or not, will never use this jump again. This implies that (cfr. Figure 1(a))

$$\begin{aligned} N_1(y, z) &\leq N(y-1, z) + N(y-1, z-1) \\ &\leq Fib(y+z) + Fib(y+z-1) = Fib(y+z+1) = Fib(x+1). \end{aligned}$$

Case 2: the first jump appears at most twice on each path. Since we have two equal jumps, the size of the jump cannot exceed the maximum number of nodes reachable using degree $y - 1$ and diameter $z - 1$ remaining after the first jump has been used (cfr. Figure 1 (b)). The part remaining after the second jump cannot exceed the maximum number of nodes reachable using the remaining degree $y - 1$ and diameter $z - 2$. Hence

$$\begin{aligned} N_2(y, z) &\leq 2N(y-1, z-1) + N(y-1, z-2) \\ &\leq 2Fib(y+z-1) + Fib(y+z-2) = Fib(x+1). \end{aligned}$$

Case 3: the first jump appears at most ℓ times on each path for some $2 < \ell \leq z$. As in Case 2, we can deduce that (cfr. Figure 1 (c)) $N_3(y, z) \leq \ell N(y-1, z-\ell+1) + N(y-1, z-\ell)$, which in turn can be shown to give

$$N_3(y, z) < Fib(y+z+1) = Fib(x+1).$$

From the above inequalities, we get

$$N(y, z) \leq \max\{N_1(y, z), N_2(y, z), N_3(y, z)\} \leq Fib(x + 1).$$

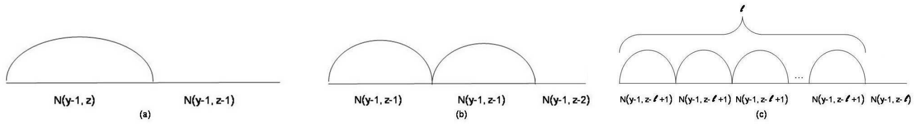

Fig. 1. The three cases in the proof of Theorem 1.

4 F-Chord: a Fibonacci-based Chord-like System

The purpose of this section is to introduce our schemes that are based on the Fibonacci number system applied to a Chord-like network, namely a ring of N identifiers labelled from 0 to $N - 1$.

The idea of using a different base (other than 2) for representing IDs (and consequently to route among them) is not new. It was already pointed out in [15], in fact, that any base $b < 2$ can be used for a Chord-like routing that trades off routing table size (i.e. degree) with number of hops (i.e. diameter). As a result, the jumps would be b^i, for each $i < \log_b N$ and, consequently, Chord's results are slightly changed: the degree is raised to $\log_b N$ while the diameter is lowered to $\log_{b/(b-1)} N$. By using a Chord-based scheme with base ϕ, one could, therefore, improve on the diameter by paying off a corresponding increase in the degree.

Nevertheless, by exploiting the many properties of Fibonacci numbers, we can define and analyze a family of uniform routing algorithms that offers improved performances on the various parameters – number of jumps, diameter, average path length, and edge congestion. Each member of the family can be obtained by tuning the parameter α, with $1/2 \leq \alpha \leq 1$. In particular, for $\alpha = 1/2$ the corresponding algorithm meets the bound in Theorem 1.

In this section, we define the families by first defining the generic Chord extension to Fibonacci numbers mentioned above. Successively we present the two families of routing schemes F-Chord.

4.1 The Fibonacci Routing Scheme Fib-Chord

For sake of simplicity, we first introduce the Fibonacci routing scheme Fib-Chord. Let $N \in (Fib(m-1), Fib(m)]$. The scheme uses $m-2$ jumps of size $Fib(i)$, for $i = 2, \ldots, m-1$. Recall that $m \approx \log_\phi N = 1.44042 \log N$.

The results of next section will imply, in particular, that, by using Fib-Chord over a set of N identifiers, the degree (i.e., the number of jumps) is $1.44042 \log N - 2$, the diameter (i.e., the number of hops) is $0.72021 \log N$, and the average path length is $0.39812 \log N$.

4.2 The Family of Routing Schemes F-Chord

Intuitively, we obtain the two families of routing schemes by taking Fib-Chord and pruning them, either starting from smaller size jumps or from larger size jumps. The pruning is realized by eliminating a certain quantity (that is related to the parameter α) of the jumps with odd indices. More formally, our family is indeed composed by two subfamilies named F_a-Chord and F_b-Chord defined below.

Definition 1. Let $Fib(m-1) < N \leq Fib(m)$ and $\alpha \in [1/2, 1]$.
a) The F_a-Chord(α) scheme uses the $\lceil \alpha(m-2) \rceil$ jumps

$$Fib(2i), \text{ for } i = 1, \ldots, \lfloor (1-\alpha)(m-2) \rfloor$$

and

$$Fib(i), \text{ for } i = 2\lfloor (1-\alpha)(m-2) \rfloor + 2, \ldots, m-1.$$

b) The F_b-Chord(α) scheme uses the $\lceil \alpha(m-2) \rceil$ jumps

$$Fib(i), \text{ for } i = 2, \ldots, m - 2\lfloor (1-\alpha)(m-2) \rfloor$$

and

$$Fib(2i), \text{ for } i = \left\lceil \frac{m - 2\lfloor (1-\alpha)(m-2) \rfloor}{2} \right\rceil + 1, \ldots, \lfloor (m-1)/2 \rfloor.$$

We will use the name F-Chord(α) whenever we want to indicate any of the two schemes F_a-Chord(α) and F_b-Chord(α). Notice here that Fib-Chord = F-Chord(1) (=F_a-Chord(1)=F_b-Chord(1)); moreover, F_a-Chord(1/2) = F_b-Chord(1/2).

5 Properties of F-Chord

In this section we investigate and bound the main parameters of the F-Chord systems.

For the rest of this section, we always assume that $Fib(m-1) < N \leq Fib(m)$ and $\alpha \in [1/2, 1]$.

5.1 Diameter

Theorem 2 *For any value of* α*, the diameter of F-Chord(*α*) is* $\lfloor m/2 \rfloor \approx 0.72021 \log N$.

Proof. Since we use the greedy strategy it is enough to consider the case $\alpha = 1/2$, i.e., when there are only jumps of even Fibonacci index.

We can prove, by induction, that after performing i jumps, the search interval is reduced to $Fib(m-2i+1)$. In fact, if $i = 1$ then we are considering a jump of size greater or equal to $Fib(m-2)$, and since $N \leq Fib(m)$ the search interval is bounded by $Fib(m) - Fib(m-2) = Fib(m-1)$. Consider the situation after t jumps. If we took a jump in the previous step, the size of the search interval is limited by $Fib(m-2t+1)$. Hence, $Fib(m-2t+1) - Fib(m-2t) = Fib(m-2t-1)$. If we did not take a jump in the previous step, the search interval is limited by $Fib(m-2t)$. Performing the jump restricts the search interval to $Fib(m-2t) - Fib(m-2t-2) = Fib(m-2t-1)$. For $i = \lfloor m/2 \rfloor$ the interval is thus reduced to $Fib(2) = 1$ and the search ends.

5.2 Average Path Length

We can precisely evaluate the average path length of F-Chord(α). Let us, first, denote by $L_1(i, m)$ (resp. $L_{1/2}(i, m)$) the load for an edge of size $Fib(i)$ on all intervals of size $Fib(m)$ when we use F-Chord(1) (resp. the F-Chord(1/2)).

The following properties can be easily obtained:

1. $L_1(m, m) = 0$;
2. $L_1(m-1, m) = Fib(m-2)$;
3. $L_1(i, m) = L_1(i, m-1) + L_1(i, m-2), i < m-2$.

Indeed, either the destination is one of the first $Fib(m-1)$ nodes or among the last $Fib(m-2)$ nodes in the interval. To go from the first part to the second part

we issue a jump of size $Fib(m-1) \neq Fib(i)$. That jump is hence not counted in $L_1(i,m)$.
Moreover, for $L_{1/2}$ it can be proved that:

4. $L_{1/2}(m-1,m) = Fib(m-2)$;
5. $L_{1/2}(m-2,m) = Fib(m-1) + Fib(m-3)$;
6. $L_{1/2}(i,m) = L_{1/2}(i,m-1) + L_{1/2}(i,m-2)$ for $i < m-2$.

We can prove the following lemma.

Lemma 1. *By using properties* **1**–**6***, it holds that:*

1) $L_1(i,m) = Fib(i-1)Fib(m-i)$ *for* $1 \leq i \leq m$.
2) $L_{1/2}(2i,m) = Fib(2i-1)Fib(m-2i) + Fib(2i+1)Fib(m-2i-1)$ *for* $2 \leq 2i \leq m-1$.

Theorem 3 *The average path length of the F-Chord(α) scheme ,is bounded by*

$$0.39812 \log N + (1-\alpha)0.24805 \log N + 1.$$

Proof. We sketch the proof for F_a-Chord(α), computation for F_b-Chord(α) is quite similar. For sake of simplicity, we assume that $N = Fib(m)$. The sum of the total link load in the network, denoted by $S_\alpha(m)$, equals the sum of the lengths of each path. It is easy to show that:

$$S_\alpha(m) = \sum_{i=1}^{m-2} Fib(i)Fib(m-i-1) + \sum_{i=1}^{\lfloor(1-\alpha)(m-2)\rfloor} Fib(2i-1)Fib(m-2i-1)$$

We have

$$\begin{aligned}
S_\alpha(m) &= \sum_{i=1}^{m-2} Fib(i)Fib(m-i-1) + \sum_{i=1}^{\lfloor(1-\alpha)(m-2)\rfloor} Fib(2i-1)Fib(m-2i-1) \\
&= \frac{1}{5}[(m-1)(Fib(m) + Fib(m-2)) - Fib(m-1)] + \\
&\quad + \sum_{i=1}^{\lfloor(1-\alpha)(m-2)\rfloor} Fib(2i-1)Fib(m-2i-1) \\
&= \frac{1}{5}[(m-1)(Fib(m) + Fib(m-2)) - Fib(m-1)] \\
&\quad + \sum_{i=3}^{\lfloor(1-\alpha)(m-2)\rfloor} Fib(2i-1)Fib(m-2i-1) + Fib(m-3) + 2Fib(m-5)
\end{aligned}$$

Hence, one can find the desired value of the average path length by observing that:

$$\frac{S_\alpha(m)}{Fib(m)} < \frac{1}{5}\left[(m-1)\left(1+\frac{1}{\phi^2}\right)-\frac{1}{\phi}\right]+\frac{(1-\alpha)(m-2)Fib(5)Fib(m-7)}{Fib(m)}+1$$

The following are immediate corollaries of the above Theorems 2 and 3.

Corollary 1 *The F-Chord(*1*) scheme (=Fib-Chord) has degree* $1.44042\log N-2$*, diameter equal to* $0.72021\log N$*, and average path length equal to* $0.39812\log N$*.*

Corollary 2 *The F-Chord*$(1/2)$ *scheme has degree and diameter both equal to* $0.72021\log N$ *and the average path length is* $0.52215\log N$*.*

We notice that F-Chord(1/2) meets the bound in Theorem 1.

Corollary 3 *For each* $<\alpha\in[0.58929, 0.69424]$*, the* F_a*-Chord(*α*) and* F_b*-Chord(*α*) schemes improve on Chord in all parameters (number of jumps, diameter, and average path length).*

5.3 Edge Congestion

We remind the reader that, by using a uniform scheme, there is no node congestion. Therefore, here we focus on edge load, that is defined as the number of times it is used by all routes from every node to every other node.

Analysis of $L_1(i,m)$ and $L_{1/2}(i,m)$ shows that $L_{1/2}(i,m)$ is decreasing with $i>1$, whereas $L_1(i,m)$ decreases until it reaches its minimum at $i=\lceil m/2\rceil$, after which it is increasing. Hence, the maximum and the minimum loads are given by

$$max\{L_1(i,m), L_{1/2}(i,m)\} = L_{1/2}(2,m) = Fib(m-1)+Fib(m-3)$$
$$min\{L_1(i,m), L_{1/2}(i,m)\} = L_1(\lceil m/2\rceil, m) = Fib(\lceil m/2\rceil -1)Fib(\lceil m/2\rceil),$$

and the ratio between the two is bounded by $9/4$, which can be viewed as the worst possible congestion. On the other hand, in the literature congestion is sometimes defined as the maximum load divided by the average load. With this definition the congestion g_α for the F-Chord(α) scheme lies between $g_{1/2}\le g_\alpha\le g_1$, and the boundaries are swiftly calculated to $g_{1/2}=1.18034$ and $g_1=1.38197$.

5.4 A comment on scalability

In the case of F-Chord(1) and F-Chord(1/2), with $n \leq N$ nodes in the network, results can be easily rewritten in terms of n with high probability, as already noticed in [15, 16]. When choosing a parameter $\alpha \in (1/2, 1)$, the presence of $n < N$ nodes will automatically tune (due to the fact that the shortest jumps will point to the same node): to $\alpha_a(n) > \alpha$ if one is using F_a-Chord(α); to $\alpha_b(n) < \alpha$ if one is using F_b-Chord(α). Thus it is possible to choose which member of the family is most suitable for the requirements of the application. If a lower average path length is desired one can choose F_a-Chord(α) (here, fixing α assures an upper bound of $\lceil \alpha(m-2) \rceil$ on the node degree for large values of n) while if a low degree is the main goal one can choose F_b-Chord(α) (in this case fixing α assures an upper bound on the average path length as in Theorem 3).

6 Conclusions and Open Problems

We have described a family of simple algorithms that *(1)* improves uniform routing on Chord and *(2)* is of practical interest. In fact, the designer can choose which member of the family is most suitable for the requirements of the application. For example, in a distributed file-system application one may want to prefer lower average path length over worst case (and thus choose F-Chord(1)), while in an application where fast delivery is paramount, one would choose a faster worst case over the average (i.e. F-Chord(1/2)) with a whole range of intermediate choices by using appropriate values of α.

Since any greedy routing requires $\Omega(\log N)$ hops when the degree is logarithmic [8], we believe that it is meaningful to improve the multiplicative constants in front of the $\log n$ since the results obtained by deterministic and uniform algorithms must be compared, practically, to the more theoretically appealing $O(\log N / \log \log N)$ that, for some values of n could have performances comparable to the deterministic and uniform algorithms (that are much much simpler to realize and deploy).

It eluded us to find an optimal average path length routing algorithm (once the node degree anf the uniformity requirements are fixed). The search schemes used in F-Chord is close to optimal but not for all values of N.

References

1. P. Druschel and A.Rowstron, "Pastry: Scalable, distribute object location and routing for large-scale peer-to-peer systems" in Proc. of the 18th IFIP/ACM Inter. Conf. on Distr. Systems Platforms (Middleware 2001), Nov. 2001.

2. Ding-Zhu Du, Frank K. Hwang. “Combinatorial Group testing and its applications”. World Scientific, 2000.
3. S. Kapoor, E.M.Reingold, “Optimum Lopsided Binary Trees”, Journal of ACM, Vol. 36, No.3, July 1989, pp. 573–590.
4. M. F. Kaashoek and D. R. Krager, “Koorde: A simple degree-optimal distributed hash table”, in IPTPS, Feb. 2003.
5. P. Ganesan, G. S. Manku, “Optimal Routing in Chord”, to appear in Proc. of SODA ’04, 2004.
6. R. Graham, O. Patashnik, D.E. Knuth “Concrete Mathematics”, Addison-Wesley, 1994.
7. A. Kumar, S. Merugu, J. Xu and X. Yu “Ulysses: A Robust, Low-Diameter, Low-Latency Peer-to-peer Network”, to appear in the Proc. of IEEE Inter. Conf. on Network Protocols (ICNP 2003), Atlanta, Nov. 2003.
8. G. S. Manku, M. Naor, U. Wieder. “Know thy Neighbor’s Neighbor: The Power of Lookahead in Randomized P2P Networks” to appear in Proc. of STOC 2004.
9. D. Malkhi, M. Naor and Ratajczak, “Viceroy: A Scalable and Dynamic Emulation of the Butterfly”, in Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC 2002), Aug. 2002.
10. M. Naor and U. Wieder, “A Simple Fault Tolerant Distributed Hash Table,” in IPTPS, Feb. 2003.
11. M. Naor and U. Wieder, “Novel architectures for p2p applications: the continous-discrete approach.” in Proc. of 15th ACM Symp. on Parallel Algorithms and Architectures (SPAA), 2003.
12. M. Naor, U. Wieder. “Know thy Neighbor’s Neighbor: Better Routing for Skip-Graphs and Small Worlds”, to appear in Proc. of IPTPS 2004.
13. S. Ratnasamy, S. Shenker, and I. Stoica, “Routing Algorithms for DHTs: Some Open Questions” in IPTPS, 2002.
14. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S.Shenker, “A scalable content-addressable network” in Proc. ACM SIGCOMM, Aug. 2001.
15. I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. Balakrishnan, “Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications”, in IEEE/ACM Trans. on Networking, 2003.
16. J. Xu, “On the Fundamental Tradeoffs between Routing Table Size and Network Diameter in Peer-to-Peer Networks”, in the Proc. of IEEE INFOCOM 2003, May 2003.
17. B. Y. Zhao, J. Kubiatowicz, and A. Joseph, “Tapestry: An infrastructure for fault-tolerant wide-area location and routing” Tech. Rep. UCB/CSD-01-1141, University of California at Berkeley, Computer Science Department, 2001.