# NoN-Greedy Routing Algorithms Are Optimal
or
Know thy Neighbor's Neighbor

Moni Naor[*†] Udi Wieder [†]

December 8, 2003

**Abstract**

Greedy routing is a common approach when a graph has some underlying metric. We present an approach for designing routing algorithms called NoN-Greedy (Neighbor-of-Neighbor). We show that in two settings it reduces significantly the number of hops taken before the destination is reached and can yield degree optimal routing (in a network of degree $O(\log n)$ this means $O(\log / \log\log n)$ hops). In particular we consider the skip-graph peer-to-peer network and the small world settings and show that both models indeed permit degree optimal routing. The theoretic results are backed by simulations which show a significant improvement. While not being Greedy per-se, the NoN-Greedy algorithm preserves some good properties of greedy algorithms. Furthermore we show a lower bound proving that in order to reduce the number of hops, information regarding neighbors only is insufficient, thus the NoN approach is essential.

## 1 Introduction

Over the last few years a very active research area has been that of "small world networks", where the phenomenon that naturally arising networks exhibit short path between pairs of nodes is investigated. Consider for instance social networks, where each node corresponds to a person and there is an edge between two nodes if the two people are acquaintances. The commonly held story about these networks is that of *six-degrees of separations*: that any two people in the world are connected via a chain of six acquaintances[1]. The behavior of such networks has been investigated extensively by researchers from diverse set of disciplines including : the social sciences, physics, computer science and webologists. These investigations consist of either checking the existence of the phenomenon in various setting or coming up with models to explain it.

The study of the algorithmic or routing perspective of this phenomenon was initiated by Kleinberg [16],[15], who pointed out that the small world experiments showed not only that short paths exist, but that *people can find such paths based on local information*. He defined a model of random graphs based on a

*Incumbent of the Judith Kleeman Professorial Chair.

†Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science. Rehovot 76100 Israel. {naor,uwieder}@wisdom.weizmann.ac.il

[1] According to Barabási [4] this idea may have its origins in a short story "Chains" by the Hungarian writer Frigyes Karinthy from 1929; this idea has been retold and recast many times since then, in the literature, popular press as well as scientific studies. The quantitative study started with Milgram's [19] experiments in 1960's asking people to send letters to unfamiliar targets only through acquaintances.

grid with additional edges. An edge between a pair of nodes is added with probability proportional to their distance squared (in the two dimensional case)[2]. For this model he showed that if a source wants to send a message to some destination, then routing the message in a *greedy* manner, i.e. each intermediate node passes the message to its neighbor that is the closest to the destination in the grid metric, yields $O(\log n)$ long paths, where $n$ is the number of nodes in the network.

Kleinberg's results can have various interpretations: it could be thought of as an explanation of how people in the chain letter experiments behaved (this is a descriptive approach). Alternatively, it could be seen as suggesting a routing strategy. While sending letters to unknown targets is not the most useful activity, the problem is related to routing in peer-to-peer networks, i.e. networks where nodes join and leave the system dynamically. In various p2p systems nodes are assigned labels that are interpreted as points on some $d-$dimensional space; links are added to close neighbors and some links are added to far away ones. Hence the hope is that lessons learned for the small world graphs may be applicable in the peer-to-peer environment. Indeed we shall show such an adaptation in this work.[3]

The greedy routing approach has many advantages, among which are the following:

*Simplicity* - greedy routing is easy to understand and implement. The routing is oblivious in the sense that the link used depends only on the destination of the message.

*Fault Tolerance* - In greedy routing closer is better. So when nodes or links fail, as long as each node has some edge towards the target, it is guaranteed that the message would reach its destination. In many constructions (such as Chord and Tapestry) it is assumed that a ring like structure exists. It is clear that as long as the ring edges exist, greedy routing would eventually succeed.

*Locality in the key space* - Messages do not 'wander' in the key space. If the source and the target are close to each other in the key space, then during the routing, the message would stay between the source and the target, in a small portion of the key space. If the key of a resource is associated with its location in the physical world then greedy routing might have proximity preserving properties; i.e. it might minimize the physical distance a message travels. If keys have *semantic* meaning, such as file names and sizes, then greedy routing would supply prefix search (as in skip graphs, see Section 3).

But is greedy routing really the best we can hope for? In terms of the degree of the network it is not optimal: for instance in the small world model of Kleinberg the average and maximum degree is logarithmic, so a priori it is not clear that we cannot find shorter paths - $O(\log n/\log\log n)$ long. In this paper we explore a variant of the greedy algorithm which we call *Neighbor-of-Neighbor Greedy*, or *NoN-Greedy*, where the optimization is over *two hops* instead of just over one. In other words, each intermediate node passes the message to its neighbor that has a neighbor that is the closest to the destination in the underlying metric.

## 1.1 Our Contributions

We consider the NoN-Greedy Algorithm over two families of probabilistic networks: one is the *small world graph* model of Kleinberg mentioned above and its various generalization to arbitrary dimensions. The other is a probabilistic peer-to-peer network called *Skip Graph* or *Network* proposed by Aspnes and Shah [3] and by Harvey *et al* [12]. While recently many proposals were made for p2p networks allowing locating of resources by names(e.g. files), Skip Graphs are the only p2p construction yet known, that offers *prefix and proximity search* on resource names (at least in a direct manner, without simulating another data structure). Skip graphs define naturally a distribution on graphs with logarithmic degrees.

[2]The model is very related to the long range percolation model studied in Physics; see Section 3.1. Also there are several related models, we chose the model as described in Coppersmith *et al* [8].

[3]The works of Aspnes *et al* [2] and Bawa *et al* [6] apply intuitions from small worlds into peer-to-peer constructions.

Our main contribution is showing that NoN-Greedy is a degree optimal routing procedure for both the skip-graphs and for the 'small-worlds' families. That is for both families we show that for any pair of source and destination Non-Greedy routing takes an $O(\log n/\log\log n)$ number of steps on the average and for the latter this holds with high probability. Thus, skip-graphs are the only construction which supports prefix search and is degree optimal. We checked the performance of the NoN-Greedy Routing Algorithm on the two families of networks discussed here and found for large sizes a saving of between $42\%$ and $48\%$ on the number of hops. Furthermore, we show that algorithms based on neighbors only, such as Greedy, *cannot* find short paths (shorter than logarithmic), thus the neighbors-of-neighbors approach is in fact essential if one wants to be degree optimal in those networks.

One question is whether the NoN-Greedy Algorithm can be implemented in the network environments we are considering. In social networks participants may use some biographical information they have about their acquaintances and hence they might be able to perform something close in nature to the algorithm (see more about it in Section 2.2.1). In peer-to-peer networks it is certainly possible to execute the NoN Greedy Algorithm, but the question is the cost, are we not squaring the degree (working on the squared graph)? The answer is no, since in such networks the cost of link stems is the fact that it should be maintained, the other side of link 'pinged' periodically etc and not from the memory for storing it. Simply having information about values your neighbors have is not that costly. Furthermore this information does not have to be up-to-date, at worse a less than optimal choice of neighbor would be made once or twice along the path, without having a disastrous affect on the route.

**Organization:** In the remaining parts of the introduction we discuss related work and present the NoN-Greedy Routing Algorithm. In Section 2 we discuss the small world model and the performance of the algorithm there and in Section 3 we discuss Skip Graphs and the performance of the algorithm on them. In Section 4 we show the simulation results for both networks. Section 5 deals with algorithms that use only neighbor information (such as the greedy algorithm) and show that in both families the performance of such algorithm is sub-optimal $\Omega(\log n)$.

## 1.2 Related Work

As mentioned above, both the areas of small world networks and p2p networks have been very active recently and we shall not attempt to even summarize the developments of the areas. For more details on the history of small world models see Section 2.1. For related lower bound on small worlds see Section 5.

One of the more influential p2p networks is Chord [23], which simulates a hypercube and has logarithmic degrees. Ganesan and Manku [11] have analyzed it and showed that the *average diameter* in Chord is $\Omega(\log n)$ and hence it is not degree optimal. Some p2p network constructions have managed to provide degree optimal routing. Viceroy [17] routes in $O(\log n)$ hops while maintaining constant degrees. Subsequently constructions based on the De-Bruijn graph ([20],[14],[10]) have provided an optimal degree-hops tradeoff for every degree and in particular for degree $\log n$ the number of hops is $O(\log n/\log\log n)$. The routing algorithms for these constructions are *not* greedy but rather are based on some numeric computation over the labels. As a consequence, even if the source and target are close in the label space, the message might be routed to far away nodes. Manku [18] suggested a randomized construction with optimal routing. While (probably) being local in the key space, the construction relies on the randomization of keys.

The basic idea of the NoN-Greedy Routing Algorithm is drawn from a paper by Coppersmith *et al* [8], which proposed the neighbors-of-neighbors approach, though not in an algorithmic perspective. They used it to show that the diameter of the 'small worlds' graph is $O(\frac{\log n}{\log\log n})$ w.h.p. The NoN-Greedy algorithm was first used by Bawa *et al* [6] as a heuristic for the Symphony peer-to-peer construction. Symphony is in fact a constructive adaptation of Kleinberg's small world model into a dynamic hash table. While not providing any asymptotical improvement, Bawa *et al* show through simulations that in practice the NoN algorithm reduced the expected latency of Symphony by $40\%$.

### 1.3 The NoN-Greedy Routing Algorithm

We now introduce the main object of our investigation, the NoN-Greedy Routing Algorithm. We assume a graph where there exists some metric on the labels of nodes. The NoN algorithm is described in Figure 1.

**Algorithm for routing a message to node $t$.**

1. Assume the message is currently at node $u \neq t$. Let $w_1, w_2, \ldots, w_k$ be the neighbors of $u$.
2. For each $w_i$, $1 \leq i \leq k$, find $z_i$ - the closest neighbor to $t$. Let $j$ be such that $z_j$ is the closest to $t$ among $z_1, z_2, \ldots, z_k$.
3. Route the message from $u$ via $w_j$ to $z_j$.

Figure 1: The NoN-Greedy Algorithm, some metric over the labels of nodes is assumed.

The Neighbor-of-Neighbor algorithm is not greedy since $w_j$ may not be the neighbor of $u$ which is closest to $t$. The algorithm could be viewed as a greedy algorithm on the *square* of the graph; i.e. the message routes through the best possible node among the nodes at distance 2.

## 2 NoN Greedy in 'Small-Worlds'

### 2.1 The Model of 'Small-World' Networks

The Small-World model originates in a classical percolation model called long range percolation. In that model edges are added between nodes on a lattice with some positive probability. The question of existence of infinite components were considered by Schulman [22], Aizenman and Newman [1] and Newman and Schulman [21], where the one dimensional lattice $\mathcal{Z}$ is studied and edges $(i, j)$ are selected with probability $\beta/|i-j|^s$ for some values $\beta, s$. A finite version of the model was suggested by Kleinberg in [16], where a two dimensional grid is studied. Each node $v$ selects $q$ neighbors, where edge $(v, u)$ is selected with probability proportional to $1/||u-v||^s$ for some parameters $s, q$. Inspired by the work of Watts and Strogatz [24], Kleinberg's goal was to model a social network. The intuition behind being that if nodes correspond to individuals, and edges correspond to social links, then it is more likely that one is acquainted with people that are 'close' to them in some sense. The main novelty in Kleinberg's work is the study of *routing* algorithms. It is shown in [16] that efficient routing is possible if $s = 2$, furthermore If $1 < s < 2$ then it is shown that any routing algorithm would take $n^\epsilon$ time, which is exponential in the expected path length. Kleinberg's results were extended to the one dimensional case by Barrière *et al* [5].

Peer-to-peer constructions based on this model were suggested by Aspnes *et al* [2] and Bawa *et al* [6]. Aspnes *et al* showed that these models have high fault tolerance and may serve as a p2p application. Bawa *et al* used Kleinberg's approach to construct a peer-to-peer system called Symphony. They showed that the parameters of the system are competitive with other p2p constructions when $q$ (the degree of the graph) is as low as $4$. In this context they introduced the NoN algorithm as a heuristic.

Coppersmith *et al* [8] studied the $d-$dimensional case where each edge $(u, v)$ is selected independently with probability $1/||u-v||^d$. They show that the *diameter* of the graph is w.h.p $\Theta(\log n/\log\log n)$. Their proof used the neighbor of neighbor approach for part of the way, and a not local argument for the rest of the way, thus their argument is not compatible with an efficient routing algorithm. We work in the model suggested in [8] and show that NoN-Greedy can be used to find short path with high probability. First let us define exactly the graphs we work with.

**Definition 2.1.** A *small world* graph of dimension $d$ is a graph whose vertex set is associated with the $d-$dimensional mesh, and each edge $u, v$ is occupied with probability $\frac{1}{||u-v||^d}$, where $||u-v||$ stands for the $L_1$ distance between $u$ and $v$.

## 2.2 Analysis of NoN-Greedy in the Small-Worlds Model

We show that for any two nodes, the NoN-Greedy succeeds w.h.p in finding a path quickly.

**Theorem 2.2.** *Using the NoN-Greedy algorithm, a message is routed between any two nodes, in* $O(\frac{\log n}{\log\log n})$ *hops with probability* $1-\frac{1}{n}$. *Where the probability is taken over the configuration of the graph.*

*Proof.* We partition the routing into two phases. In the first phase the message is routed to a distance of $e^{\sqrt{\log n}}$ from the target and in the second phase the message is routed the remaining distance. Each phase would take $O(\log n/\log\log n)$ w.h.p thus proving the theorem. The first phase was handled in Lemma (6.1) from [8].

**Lemma 2.3 ([8]).** *If* $m = (2d+2)\cdot 2^{d+1}\log n/\log\log n$, *then after* $m$ *routing steps the message would be at distance of at most* $e^{\sqrt{\log n}}$ *from the destination, with probability at least* $1-\frac{1}{n^{2d}}$.

The second phase of the routing could in fact be performed by Greedy.

**Lemma 2.4.** *Assume a message is at distance* $e^{\sqrt{\log n}}$ *from its destination. With probability* $1-\frac{1}{n^{2d}}$ *the message would reach its destination after* $O(\log n/\log\log n)$ *greedy steps.*

First we show the following:

**Claim 2.5.** *Assume the message is at distance* $n$ *from the destination and after performing one greedy step the message is at distance* $n'$. *There is an* $\epsilon > 0$ *independent of* $n$, *such that*
$\Pr[n' \leq \lceil(1-\frac{1}{k})n\rceil] \geq 1-\frac{1}{k^\epsilon}$.

*Proof.* Assume the message is at node $\vec{0}$, and the target note $t$ is such that $||t|| = n$. For each $k$ define $B_k$ to be all nodes with distance at most $(1-\frac{1}{k})n$ from $t$ (for convenience we remove the ceilings and floors). We calculate the probability there is an edge from $\vec{0}$ to the ball $B_k$. Define $\ell_i$ to be the number of vertices $x$ such that $||x|| = i$ and $x$ is in $B_k$. We have:

$$\Pr[\vec{0} \text{ is not connected to } B_k] = \prod_{i=n/k}^{n}(1-i^{-d})^{\ell_i} \leq \prod_{i=2n/k}^{n}(1-i^{-d})^{\ell_i} \leq \prod_{i=2n/k}^{n} e^{\ell_i/i^d} = \exp(\sum_{i=2n/k}^{n}\frac{\ell_i}{i^d})$$

Now assuming that $\ell_i$ is $\Theta(i^{d-1})$ for $\frac{2n}{k} \leq i \leq n$, for some constant $\epsilon$ it holds that

$$\exp(\sum_{i=2n/k}^{n}\frac{\ell_i}{i^d}) \leq \frac{1}{k^\epsilon}$$

which proves the claim. It remains to show that indeed $\ell_i = \Theta(i^{d-1})$ for $\frac{2n}{k} \le i \le n$. There are $\Theta(i^{d-1})$ nodes at distance $i$ from $\vec{0}$. we need to show that a constant fraction of them are in $B_k$. Let $i$ take some value $2n/k \le i \le n$. Now let $x$ be a point on a shortest path from $\vec{0}$ to $u$ such that $||x|| = \frac{3}{4}i$; i.e. $x$ is at distance $n - \frac{3}{4}i$ from $u$. There are $\Theta((\frac{1}{4}i)^{d-1}) = \Theta(i^{d-1})$ points that are of distance $\frac{1}{4}i$ from $x$. How many of them are of distance $i$ from $\vec{0}$? The points at distance $\frac{1}{4}i$ from $x$ are evenly divided between the $2^d$ quadrants of the ball around $x$. It follows that a $2^{-d} = \Theta(1)$ fraction of them are at distance $i$ from $\vec{0}$. The distance of each of these points from $u$ is at most

$$(n - \frac{3}{4}i) + \frac{1}{4}i = n - \frac{1}{2}i \le n(1 - \frac{1}{k}).$$

Which concludes the proof of Claim 2.5. □

*Proof of Lemma 2.4.* According to the previous claim, the probability the distance is reduced by a factor of $1-\frac{1}{(\log n)^{1/4}}$ is $1-\frac{1}{\log^{\epsilon} n}$. This means that $o(\log n/\log\log n)$ steps, each reduces the distance by $1-\frac{1}{(\log n)^{1/4}}$, would route the message to the destination. We prove this occurs with probability $1-\frac{1}{n^{2d}}$ using the following argument: Let $X_i$ be the random Bernoulli variable indicating whether the $i^{th}$ NoN-hop have failed in reducing the distance by a factor of $1 - \frac{1}{(\log n)^{1/4}}$. We know that $\Pr[X_i = 1] \le \frac{1}{\log^{\epsilon} n}$. Now assume that the variable $X_i$ is simulated by tossing $\epsilon \log\log n$ fair coins and setting $X_i = 1$ if all coins turned up to be 1. Now we have $c \log n$ fair coins, and if less than $\frac{3}{4}$ of the coins turned up to be 1 the algorithm will not fail. The standard Chernoff bound [7] shows there is a constant $c$ such that this happens with probability at least $1 - \frac{1}{n^{2d}}$.

□

The proof of Theorem 2.2 is now completed by combining Lemma 2.3 which handled the first phase of the routing, with Lemma 2.4 which handled the second phase of the routing.

□

#### 2.2.1 Do People Use the NoN-Greedy Algorithm in Social Networks?

Since the original motivation of analyzing small-world graphs was the modelling of social networks, it is interesting to check whether people use the NoN-Greedy algorithm when they are required to navigate in a social network. Recently Dodds *et al* [9] repeated the famous experiment by Milgram [19], in which letters were passed between random nodes on a social network where edges corresponds to say a first name acquaintance. In [9] people were given a target and were asked to send an email to a person they are acquainted with, where the goal is that the email would reach its destination quickly. People were also asked *why* they forwarded the mail to the person whom they have chosen. It appears that in the first two steps of the 'routing', which are most meaningful, about 25% of the people sent the message to a recipient for one of the following reasons:

1. The recipient has travelled to target's geographical region.
2. The recipient's family originates from the target's geographical region.

Both these reasons imply that the recipient received the message based on who his (possible) friends are, and not on the individual characteristics of the recipient himself. Other reasons such as - the recipient has the same education as the target - could be viewed both as Greedy and NoN-Greedy steps. We can conclude that at least some of the time the NoN-Greedy algorithm was used.

## 3 P2P Application - Improving Skip Graphs

In this section we show that NoN-Greedy routing reduces the expected latency of searches in a known p2p application - Skip Graphs. A Skip Graph or a Skip Net is an implementation of a dynamic distributed hash table, which was suggested by Aspnes and Shah in [3] and by Harvey *et al* in [12]. It is meant to serve as a mean for locating resources in a p2p network. We follow the description in [3].

### 3.1 The Skip Graph Model

In a Skip Graph each node represents a resource to be searched. Node $x$ holds two fields: the first is a key - $k(x)$ which is arbitrary and might be the resource name. Nodes are searched according to their keys. We assume for notational convenience that the keys are the integers $1, 2, \dots, n$. The second field is a membership vector $m(x)$ which is an infinite string of *random bits* chosen independently by each node. We think of $|m(x)|$ as infinite for convenience, actually $O(\log n)$ bits suffice with overwhelming probability. The nodes are ordered lexicographically by their keys and placed on the line so node $i$ is connected to $i-1$ and $i+1$. The edges are determined by the membership vectors in the following way. Denote by $m(x)_k$ the first $k$ bits of $m(x)$. Nodes $x, y$ are connected by an edge if there exists some $k$ such that $m(x)_k = m(y)_k$, and for each $x < z < y$ it holds that $m(z)_k \neq m(x)_k$. In other words, two nodes are connected by an edge if their corresponding membership vectors share some prefix which is not shared by any of the nodes between them.

It is easy to see that w.h.p all nodes have a logarithmic degree. The main advantage of skip graphs is that if the desired key is not found, it is easy to find the key closest to it in the *key space*. Since keys are arbitrary and may represent resource names, this operation has a semantic meaning. This is in contrast with previously known P2P constructions that achieved $O(\log n/\log\log n)$ routing for logarithmic degree([20],[14],[10],[18]) which relied on the randomization of the key space, and therefore closeness in the key space carries no semantic meaning.

### 3.2 Routing in Skip Graphs

The routing algorithm suggested in [3],[12] routes the message using the longest prefix of $m(x)$ possible, without overshooting the target. In other words it is a greedy algorithm which moves the message as close to the target as possible without overshooting, and routes in $O(\log n)$ hops. We improve by showing in Theorem 3.1 that the NoN-Greedy algorithm routes in $O(\log n/\log\log n)$ hops, and by showing in Theorem 5.1 that the Greedy algorithm needs $\Omega(\log n)$ time to route. The NoN-Greedy algorithm would consider two consecutive hops and would try to route the message as close as possible to the target.

**Theorem 3.1.** *The expected length of a path between node* $0$ *and node* $n$ *is* $O(\log n/\log\log n)$*, where the expectation is taken over the choices of membership vectors.*

*Proof.* We show that with a fixed probability there exists a path of length 2 between 0 and the segment $[(1-\frac{1}{\log n})n, n]$; when such a path exists we say the algorithm succeeds. This means that in expectation it would take the algorithm $O(\log n/\log\log n)$ attempts until it succeeds $\Theta(\log n/\log\log n)$ times, which is sufficient to prove the theorem.

Let $A_{ij}$ be the event: "there is an edge between $i$ and $j$, and this edge corresponds to a prefix of length $\lfloor \log|i-j| \rfloor$". The restriction on the length of the prefix which induces is the edge is done for technical reasons which would be cleared shortly. We may write $A_{ijk}$ meaning $A_{ij} \wedge A_{jk}$.

**Lemma 3.2.** *For each* $0 \leq i, j \leq n$ *we have* $\Pr[A_{ij}] = \Theta(\frac{1}{|i-j|})$.

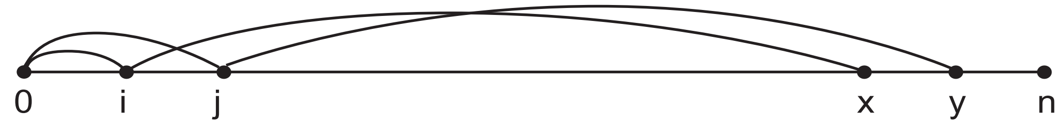


Figure 2: The events $A_{0ix}, A_{0ij}$.

*Proof.* Consider the probability $i, j$ share a prefix of length $k$ while the $|i-j|-1$ nodes between them do not share it. This probability is $2^{-k} \cdot (1-2^{-k})^{|i-j|-1}$. Now if $k = \lfloor \log(|i-j|) \rfloor$ then there exists two constants $c_1, c_2$ such that

$$\frac{c_1}{|i-j|} \leq 2^{-k} \cdot (1-2^{-k})^{|i-j|-1} \leq \frac{c_2}{|i-j|}$$

which proves the lemma. $\square$

Let $X$ denote the number of paths of length 2 that connect node 0 with the segment $[(1-\frac{\alpha}{\log n})n, n]$. Limit the first of the two steps to be of length at most $n/4$ and the second of the two steps to correspond to a prefix of length $\lfloor \log n \rfloor$. Our goal is to show that with a fixed probability $X$ is greater than zero. We start by calculating its expectation.

**Lemma 3.3.** *There exists a constant $\alpha$ such that $\mathbb{E}[X] \geq 5$.*

*Proof.* Let $V$ denote the set of nodes in the segment $[(1-\frac{\alpha}{\log n})n, n]$. By linearity of expectation we have

$$\mathbb{E}[X] = \sum_{i<n/4} \sum_{k \in V} \Pr[A_{0ik}] = \sum_{i<n/4} \sum_{k \in V} \Pr[A_{0i} \wedge A_{ik}] = \sum_{i<n/4} \sum_{k \in V} \Pr[A_{0i}] \cdot \Pr[A_{ik}]$$

where the last equality holds since $A_{ij}$ and $A_{jk}$ are independent. For a fixed $k \in V$ Lemma 3.2 shows that

$$\sum_{i<n/4} \Pr[A_{0i}] \cdot \Pr[A_{ik}] \geq c_1 (\sum_{i<n/4} \frac{1}{i(k-i)}) \geq c_2 (\sum_{i<n/4} \frac{1}{i(n-i)}) \geq c_3 (\frac{\log n}{n}).$$

Now $\mathbb{E}[X] \geq \alpha c_3$ and the lemma holds for $\alpha$ a large enough constant. $\square$

By Chebyschev's inequality $\Pr[X=0] \leq \frac{var[X]}{\mathbb{E}^2[X]}$. We show that $\Pr[X=0] \leq 0.7$ by bounding the size of the covariance of each pair $A_{0ix}, A_{0jy}$.

**Lemma 3.4.** *For each $0 < i, j < n/4$ it holds that $cov[A_{0ix}, A_{0jy}] \leq \frac{1}{2} \Pr[A_{0ix}] \cdot \Pr[A_{0jy}]$.*

*Proof.* Consider a pair of events as depicted in figure 2. We have:

$$\Pr[A_{0ix} \wedge A_{0ij}] = \Pr[A_{0i}] \cdot \Pr[A_{0j} | A_{0i}] \cdot \Pr[A_{jy} | A_{0i}, A_{0j}] \cdot \Pr[A_{ix} | A_{jy}, A_{0i}, A_{0j}].$$

Bound the size of each of the elements in the expression. The occurrence of the event $A_{0i}$ means that $m(i)$ could not be used in order to establish an edge $(0, j)$, therefore $\Pr[A_{0j} | A_{0i}] \leq \Pr[A_{0j}]$. The event $A_{jy}$ is independent of the membership vectors in the segment $[0, j]$ and therefore is independent of $A_{0i}, A_{0j}$. Let $\ell = \lfloor \log n \rfloor$ and $m(i)_\ell$ denote the first $\ell$ bits of $m(i)$. Consider, as before, only the case in which the second hop corresponds to prefixes of length $\ell$. Now we have that $\Pr[A_{ix} | A_{0i}, A_{0j}, A_{jy}]$ is the probability $m(x)_\ell = m(i)_\ell$, and that for each node $k$ between $i$ and $x$ it holds that $m(k)_\ell \neq m(i)_\ell$, conditioned on the existence of the edges $(0,i),(0,j),(j,y)$. The existence of the edge $(j,y)$ means that for all nodes $j<k\le x$ it holds that $m(k)_\ell \ne m(j)_\ell$; i.e. it excludes the existence of one prefix in the segment $[j,x]$ and therefore has only a small effect on the probability of $A_{ix}$. The existence of the edge $(0,j)$ is independent of membership vectors in the segment $[j+1,x]$, $j-i\le n/4$ therefore we have:

$$\begin{aligned}\Pr[A_{ix}|A_{0i},A_{0j},A_{jy}] &\le \frac{1}{2^\ell-1}\cdot\left(1-\frac{1}{2^\ell-1}\right)^{(x-j-1)}\\ &\le 1.1((1-2^{-\ell})^{i-j})\cdot(2^{-\ell}\cdot(1-2^{-\ell})^{x-i+1})\\ &\le 1.1((1-2^{-\ell})^{-n/4})\cdot\Pr[A_{ix}]\\ &\le 1.5\Pr[A_{ix}]\end{aligned}$$

We have

$$\Pr[A_{0ix}\wedge A_{0ij}]\le 1.5\Pr[A_{0i}]\Pr[A_{ix}]\Pr[A_{0j}]\Pr[A_{jy}]=1.5\Pr[A_{0ix}]\cdot\Pr[A_{0jy}].$$

$cov[A_{0ix},A_{0jy}]=\Pr[A_{0ix}\wedge A_{0jy}]-\Pr[A_{0ix}]\cdot\Pr[A_{0jy}]\le\frac{1}{2}\Pr[A_{0ix}]\cdot\Pr[A_{0jy}]$. The cases in which $x>y, x=y, i=j$ are done in an analogous way. □

Now calculate the variance -

$$var[X]=\sum_{i,x}var[A_{0,i,x}]+\sum_{i,j,x,y}cov[A_{0ix},A_{0jy}]\le\mathbb{E}[X]+\frac{1}{2}(\sum_{i,x}\Pr[A_{0ix}]\cdot\sum_{j,y}\Pr[A_{0ix}])=\mathbb{E}[X]+\frac{1}{2}\mathbb{E}^2[X].$$

Now we have

$$\Pr[X=0]\le\frac{\mathbb{E}[X]+\frac{1}{2}\mathbb{E}^2[X]}{\mathbb{E}^2[X]}\le 0.7$$

where the last inequality follows since $\mathbb{E}[X]\ge 5$. Now complete the proof of theorem 3.1. Assume the message is at distance $n$ from the target. Each time the algorithm performs a NoN-Greedy step it reduces the distance to the target by a factor of $\frac{\alpha}{\log n}$ with probability $\ge 0.3$. It would take the algorithm on average less than $10\log n/\log\log n$ steps to succeed $3\log n/\log\log n$ times. As long as the distance to the target is larger than $2^{\sqrt{\log n}}$, each success would reduce the distance by a factor of $\frac{\alpha}{\sqrt{\log n}}$. Since

$$n\cdot\left(\frac{\alpha}{\sqrt{\log n}}\right)^{\frac{3\log n}{\log\log n}}\le 2^{\sqrt{\log n}}.$$

the message would reach a distance of $O(2^{\sqrt{\log n}})$ from the target. From here on the distance to the target is reduced by a factor of $\frac{1}{2}$ with probability at least $\frac{1}{4}$. Therefore the message routes through the remaining $2^{\sqrt{\log n}}$ in less than $4\sqrt{\log n}$ hops on expectation. Conclude that the expected number of hops it takes a message to route from 0 to $n$ is at most $20\log n/\log\log n+4\sqrt{\log n}=O(\log n/\log\log n)$. □

While the constant coming out of Theorem 3.1 is 20 which is rather large, simulations have shown that for $n=2^{17}$, NoN-Greedy improves over Greedy in a factor of about 1.9, suggesting the real constant is much lower.

### 3.3 Randomization Reduces Latency

A common paradigm in the design of p2p structures is to first choose some static graph with some good qualities and then design the dynamic system such that it resembles the static one as much as possible. In case of skip graphs a 'perfect' skip graph has the $i^{th}$ edge of each node cover a distance of $2^i$; i.e. the lengths of edges of a node form a geometric series. The randomization involved in the dynamic construction is usually considered as a negative byproduct and much effort is put in reducing it. For instance a deterministic protocol that can guarantee that the skip graph is almost 'perfect' is presented in [13] and general techniques for other p2p constructions appear in [20].

As was noticed by Harvey *et al* [12], a perfect skip graph is similar to Chord [23], and therefore according to Ganesan and Manku [11] has *average diameter* of $\Omega(\log n)$. This leads to the following counter intuitive and surprising fact:

The randomization of skip graphs actually *reduces* the average path length.

The reason is that the randomization enables a routing algorithm to use an 'exceptionally' long edge once in a while. The density of these long edges being just large enough so that the NoN-Greedy algorithm finds them. In a 'perfect' skip graph, Chord, and in the hypercube - these long edges do not exist. Our results show that safety has a price: while these protocols have guaranteed bounds, they enlarge the expected latency.

### 3.4 P2P Nugget - Improving Chord

Does NoN-Greedy improve more p2p systems? We show that a variation of Chord [23] is degree optimal. In Chord the key space is a ring $[0, 1, \ldots, N]$. A node whose i.d. is $x$ is connected (more or less) to $x + 1, x + 2, x + 4, \ldots, x + \frac{N}{2}$. As noted, since the *average* diameter of Chord is $\Theta(\log n)$, no algorithm may significantly reduce the path length. We show a slight variation of Chord, (in search of a better name lets call it NoN-Chord). The idea is to make Chord resemble the Small-World graph. Now each node $x$ is connected to $\log N$ nodes $y_0, y_1, y_2, \ldots$ such that $y_i$ is a *random* point in the segment $[x + 2^i, x + 2^{i+1}]$. An easy adaptation of the previous proofs show that w.h.p the path length used by NoN is $O(\log n / \log \log n)$. The Insert operation now, would take $\log^2 n / \log \log n$ operations, in $\log n / \log \log n$ parallel time. So we have an interesting tradeoff of parameters. An increase in the communication complexity of the Insert operation (though not in the time complexity) reduces the latency of the paths.

## 4 Simulation Results

We ran simulations in which we compared the performance of the greedy algorithm and the performance of the NoN greedy algorithm. We constructed a skip graph of up to $2^{17}$ nodes. For each size we routed between 30 random sources and destination. The whole process was repeated 5 times. Thus the results shown in Figure 3 are the average of 150 executions. A substantial improvement could be seen. When $N = 2^{17}$ the number of hops is reduced from 13.7 to 7.2 which is a factor of $47.5\%$.

In a small world graph it is not necessary to create the full graph in advance. Each time the message reached a node, we randomly created the neighborhood of radius 2 around the node. This is a negligible compromise over the definition of the model, since the edge in which the node was entered might not be sampled. This technique allowed us to run simulations on much larger graphs. In Figure 4, 5 are the results of simulations for the one and two dimensional small world graphs with up to $2^{24}$ nodes. As in skip graphs, we see a substantial improvement of performance. When the number of nodes is $2^{24}$, the average number of hops is reduced from $11.7$ to $7.8$ in the one dimensional case and from $12.2$ to $9.6$ in the two dimensional case. This is compatible with the simulations done by Bawa *et al* [6], for Symphony - which is a variant of the small world graph. They showed that even when the degree is not logarithmic, but fixed to be small, the NoN algorithm reduced the number of hops by about $40\%$.

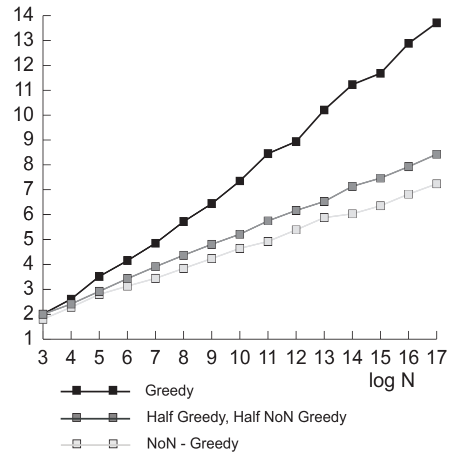


Figure 3: The number of hops in skip graphs.

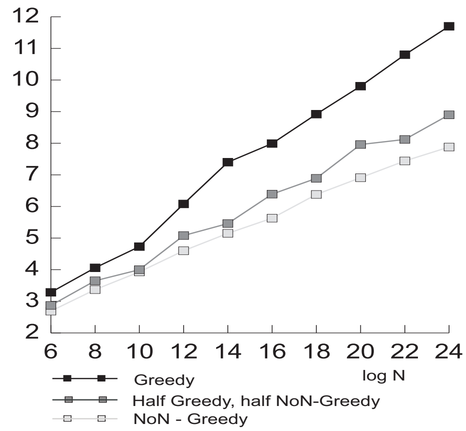


Figure 4: The number of hops in a small world of dimension 1.

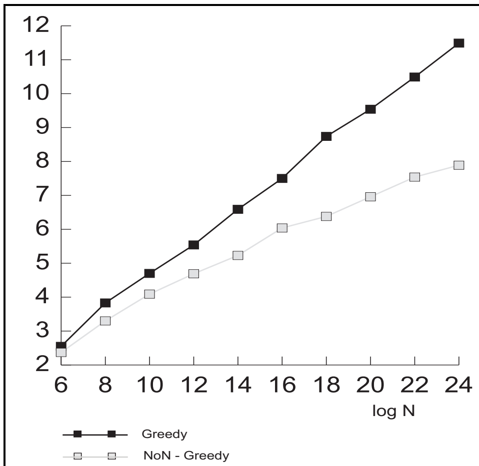


Figure 5: The number of hops in a small world of dimension 2.

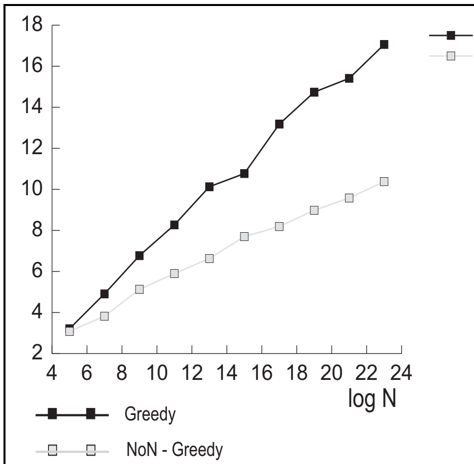


Figure 6: Small world of dimension 1, with half of the edges.

What if not all the hops are NoN hops but rather some hops are greedy? We simulated the routing when each node with probability $\frac{1}{2}$ performs a greedy step, and with probability $\frac{1}{2}$ performs a NoN step (i.e. routes through a neighbor to a neighbor's neighbor). This simulates the situation in which some of the nodes fail to hold the neighbor lists of their neighbors. We see in Figures 3 and 4 for small worlds and skip graphs, that even if just half of the nodes use the NoN algorithm , the improvement is still substantial. After that we tested the algorithm when less edges appear. We changed the model such that if $|u - v| > 1$ then the edge $(u, v)$ exists with probability $\frac{1}{2|u-v|}$. Figure 6 shows that the number of hops was reduced from 17.06 to 10.38.

We conclude that there are no large constants hidden in the proofs of Theorems 3.1 and 2.2. Indeed even for moderate network sizes, the NoN algorithms performs substantially better than the Greedy one.

### 4.1 Implementation Issues

The execution of a NoN hop requires a node to store the neighbor lists of its neighbors. This implies that nodes should update each other regarding their own lists of neighbors. Such an update occurs in two scenarios:

1. Each node upon entrance, must send its list of neighbors to its neighbors.
2. Whenever a node encounters a change in its neighbor list (due to the entrance or exit of a node), it should update its neighbors.

These updates impose some extra communication. This communication is not a heavy burden on the system for the following reasons:

1. Each node periodically checks that its neighbors are alive (for instance by pinging them). Checking whether a neighbor list has changed could be added to the maintenance protocol by sending a hash of the neighbor list.
2. The updates are not urgent and may be done when the system is not busy. Figures 3 and 4 show that even if half of the nodes are not updated, the performance of the algorithm is hardly compromised. This issue is discussed in the next section.

**Doesn't NoN require routing tables of size $\log^2 n$?** No. In order to apply NoN, the *memory* allocated to hold the routing tables of the neighbors is $O(\log^2 n)$ instead of $O(\log n)$. The main cost of an entry in the routing table lies in its *maintenance* (pinging periodically etc.). The cost in terms of memory of simply holding the entry is marginal.

**What if the neighbor's tables are not updated fast enough?** It might be the case that a node does not hold its neighbor's neighbor lists. Furthermore, if the extra lists are not updated fast enough, it might be the case that a node receives a message which it is supposed to move to a neighbor that no longer exists. In all these cases a node should simply perform a greedy step. Thus, the performance of NoN depends on how updated are the lists. If they are not updated or even do not exist at all then all the hops are greedy and the message reaches its destination after $O(\log n)$ hops. If only half of the hops are NoN hops, then after $O(\log n / \log\log n)$ hops on expectation the message would reach its destination. Figure 3 and 4 demonstrates that even if only one half of the nodes perform the NoN step, the compromise in performance is marginal, and latency is still reduced substantially.

We conclude that implementing NoN has very little cost. The extra communication required could be done when the system is free, and if for some reason the updates did not occur, then greedy routing is an excellent default.

## 5 Lower Bounds

In this section we prove that in order to find a path between nodes at distance $n$, a routing algorithm must; either run in $\Omega(\log n)$ time w.h.p (i.e. $\Omega(\log n)$ hops), or must use additional knowledge about the neighbor's neighbors of a node. The lower bound holds for a model which generalizes the Greedy algorithm, this it applies for a larger family of algorithms which includes Greedy. It holds both for small world graphs and skip graphs models (i.e. the probability space they define).

A logarithmic lower bound for the one dimension small world graph could be found in the work of Barrière *et al* [5]. Aspnes *et al* [2] extended the model to a larger family of random graphs. They show that if the average degree is $O(\log n)$ then greedy routing would take $\Omega(\log n)$ hops on average. The proof however is limited to the case where the nodes are set on a one dimensional line and the probability upon the edges has some symmetry assumptions that do not apply to skip graphs.

## 5.1 A Probing Model

Assume our goal is to find a path between two specific vertices of distance $n$, say node $0$ and node $n$. In order to do so an algorithm must *probe* the vertices of the graph, where the probing of a vertex reveals all the edges connected to it. The lower bound we show apply in a *probing* model, where we bound the number probes needed to find a path. Clearly a lower bound on the number of probes needed by the algorithm is a lower bound on the (sequential) time complexity of a routing algorithm.

Define a $1-$local algorithm to be a probing algorithm with the following properties:

1. The algorithm begins by probing the node $0$.

2. The algorithm only probes nodes to which it has already established a path from $0$.

The term *local* derives from the assumption that the algorithm starts at $0$ and is only allowed to probe nodes it has already reached. The term $1-$local is used, since the probing of a node reveals its neighborhood of radius $1$, i.e. its neighbors. If it is assumed that a probe reveals a neighborhood of radius $k$ then the algorithm is termed *$k$-local*. Every routing algorithm which relies on local information only, corresponds to a $1-$local probing algorithm. The Greedy algorithm therefore is $1-$local. The NoN-Greedy algorithm could be viewed, following Theorems 2.2,3.1 as either a $2-$local algorithm with $O(\log n/\log\log n)$ probes on average, or as a $1-$local algorithm with a probing complexity of $O(\log^2 n/\log\log n)$ on average.

**Theorem 5.1.** *(A) In a skip graph - any* 1-local *algorithm that outputs a path between two nodes at distance $n$, must probe $\Omega(\log n)$ probes, with probability $1-\frac{1}{n^\epsilon}$. In particular the expected number of probes is $\Omega(\log n)$.*
*(B) In a $d-$dimensional small world graph - any* 1-local *algorithm that outputs a path between two nodes at distance $n$, must probe $\Omega(\log n)$ probes, with probability $1-\frac{1}{n^\epsilon}$. In particular the expected number of probes is $\Omega(\log n)$.*

The theorem implies that if a node holds only its neighbors then *any* routing algorithm would need $\Omega(\log n)$ probes w.h.p. Thus the assumption that nodes have some knowledge of their neighbor's neighbors is essential.

First we argue that the Greedy algorithm dominates any other $1-$local algorithm. The following lemma holds both for skip graphs and small world graphs.

**Lemma 5.2.** *Let $A$ be some $1-$local algorithm . Denote by $f_A(n), f_G(n)$ the random variable representing the number of probes it takes the algorithm $A$ and the Greedy algorithm respectively, to find a path between two nodes at distance $n$. For all $n > k > 0$ it holds that $\Pr[f_G(n) \le k] \le \Pr[f_A(n) \le k]$.*

*Proof Sketch:* For convenience we label the source node $n$ and the target node $0$. The trick is to give $A$ some extra information. Assume that at step $i$ the closest node to $0$ which $A$ had found is at distance $d_i$

from 0, where the distance is measured by the $L_1$ norm. At this point we give $A$ all the information possible regarding nodes outside a ball of radius $d_i$ from 0. This would include all the membership vectors in the case of skip graphs, and all the edges in the case of a small world graph. Clearly this would only reduce the expected number of probes until 0 is found. Now if $n_1 > n_2$ then for every configuration of edges, every move $A$ can do in $n_1$ is also available in $n_2$, i.e. $f_A(n_1) \geq f_A(n_2)$. Conclude that for every $k$, $\Pr[f_A(n_1) > k] \geq \Pr[f_A(n_2) > k]$. Now, in this setting, a move to the closest node possible to 0, may only increase the probability that the 0 is found within $k$ steps. Both in skip graphs and in small world graphs, this is achieved by probing the node closest to 0; i.e. by the greedy algorithm. However the greedy algorithm does not require the extra help, thus the lemma is proved. $\square$

Divide the nodes of the graph into sets $B_0, B_1, \ldots, B_{\log n}$ according to their distance from 0, such that $B_i = \{u | 2^{i-1} \leq dist(u,0) < 2^i\}$. So $0 \in B_0$ and $n \in B_{\lceil \log n \rceil}$. For each $0 \leq i \leq \log n$ let $X_i$ be the indicator of the event: "The path taken by Greedy includes at least one vertex in $B_i$". Clearly the number is of nodes in the path is at least $\sum_{i=o}^{\log n} X_i$.

**Lemma 5.3.** *Both for skip graphs and for small world graphs and for each* $i$*, it holds that*

$$\Pr[X_i = 1 | X_{i+1}, X_{i+2}, \ldots, X_{\log n}] \geq c$$

*for some constant* $c$ *independent of* $n$*.*

Before proving the lemma we show why it suffices to prove Theorem 5.1. Let $Y_i$ be a Bernoulli variable with $\Pr[Y_i = 1] = c$. Now $\mathbb{E}[\sum Y_i] = c \log n \leq \mathbb{E}[\sum X_i]$. Furthermore the random variable $\sum X_i$ dominates the random variable $\sum y_i$. We have

$$\Pr[\sum X_i \leq \frac{1}{2} c \log n] \leq \Pr[\sum Y_i \leq \frac{1}{2} c \log n] \leq \frac{1}{n^\epsilon}$$

according to Chernoff's bounds.

*Proof of Lemma 5.3.* Assume the values of $X_{i+1}, X_{i+2}, \ldots, X_{\log n}$ are already set and $j$ is the smallest index such that $X_j = 1$. Clearly $X_i$ is independent from $X_{j+1}, X_{j+2}, \ldots, X_{logn}$, it remains to analyze $\Pr[X_i = 1 | X_{i+1} = 0, X_{i+2} = 0, \ldots, X_j = 1]$. Let $y$ be the node in $B_j$ which was last probed by Greedy. The notation $y \sim B_i$ stands for the event - $y$ is connected to $B_i$. We have:

$$\begin{aligned} & \Pr[X_i = 1 | X_{i+1} = 0, X_{i+2} = 0, \ldots, X_j = 1] = \\ & \frac{\Pr[X_i = 1, X_{i+1} = 0, \ldots, X_j = 1]}{\Pr[X_{i+1} = 0, \ldots, X_j = 1]} \geq \\ & \frac{\Pr[y \sim B_i \wedge y \not\sim B_{i-1} \cup \ldots \cup B_0]}{\Pr[y \sim B_i \cup \ldots \cup B_0]}. \end{aligned}$$

Let $t = \Pr[y \sim B_i]$, $t' = \Pr[y \sim B_{i-1} \cup \ldots \cup B_0]$. In case of a small world graph We have:

$$\Pr[X_i = 1 | X_{i+1} = 0, X_{i+2} = 0, \ldots, X_j = 1] = \frac{t(1-t')}{t(1-t') + (1-t)t' + tt'}.$$

It is easy to verify that $t = \Theta(t')$ and that $t' \leq \frac{3}{4}$. Under these conditions

$$\Pr[X_i = 1 | X_{i+1} = 0, X_{i+2} = 0, \ldots, X_j = 1] \geq c$$

for some constant $c > 0$ independent of $n$. The case of skip graph is almost similar. In a skip graph the events $y \sim B_i$ and $y \not\sim (B_{i-1} \cup \ldots \cup B_0)$ are not independent but are negatively correlated, so the analysis with the necessary corrections follows. $\square$

### Acknowledgments

We thank James Aspnes and Gauri Shah for supplying us with their implementation of skip graphs.

## References

[1] M. Aizenman and C. M. Newman. Discontinuity of the percolation density in one dimensional $1/|x-y|^2$ percolation models. *Communications in Mathematical Physics*, 107:611–647, 1986.

[2] J. Aspnes, Z. Diamadi, and G. Shah. Fault-tolerant routing in peer-to-peer systems. In *Proceedings of the twenty-first annual symposium on Principles of distributed computing (PODC)*, pages 223–232, 2002.

[3] J. Aspnes and G. Shah. Skip graphs. In *fourteenth ACM SIAM Symposium on Discrete Algorithms (SODA)*, pages 384–393, 2003.

[4] A. L. Barabási. *Linked: The New Science of Networks*. Perseus Publishing, 2002.

[5] L. Barrière, P. Fraigniaud, E. Kranakis, and D. Krizanc. Efficient routing in networks with long range contacts. In *15th International Symposium on Distributed Computing (DISC '01)*, pages 270–284, 2001.

[6] M. Bawa, G. S. Manku, and P. Raghavan. Symphony: Distributed hashing in a small world. In *4th USENIX Symposium on Internet Technologies and Systems, (USITS)*, 2003.

[7] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Math. Stat.*, 23:493–509, 1952.

[8] D. Coppersmith, D. Gamarnik, and M. Sviridenko. The diameter of a long-range percolation graph. *Random Structures and Algorithms*, 21(1):1–13, 2002.

[9] P. S. Dodds, M. Roby, and D. J. Watts. An experimental study of search in global social networks. *Science*, 301:827–829, 2003.

[10] P. Fraigniaud and P. Gauron. The content-addressable network d2b. Technical Report LRI 1349, Univ. Paris-Sud, 2003.

[11] P. Ganesan and G. S. Manku. Optimal routing in chord. In *fifteenth ACM SIAM Symposium on Discrete Algorithms (SODA)*, 2004.

[12] N. Harvey, J. Dunagan, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet: A scalable overlay network with practical locality properties. In *4th USENIX Symposium on Internet Technologies and Systems, (USITS)*, 2003.

[13] N. Harvey and J. I. Munro. Deterministic skipnet. In *Twenty Second Annual ACM Syposium on Priciples of Distributed Computing (PODC)*, pages 152–153, 2003.

[14] M. F. Kaashoek and D. R. Karger. Koorde: A simple degree-optimal distributed hash table. In *Second International Workshop on Peer-to-peer Systems IPTPS*, 2003.

[15] J. Kleinberg. Navigation in a small world. *Nature*, 406:845, 2000.

[16] J. Kleinberg. The Small-World phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC)*, 2000.

[17] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proceedings of the twenty-first annual symposium on Principles of distributed computing (PODC)*, pages 183–192, 2002.

[18] G. S. Manku. Routing networks for distributed hash tables. In *22nd ACM Symposium on Principles of Distributed Computing, (PODC)*, 2003.

[19] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[20] M. Naor and U. Wieder. Novel architectures for p2p applications: the continuous-discrete approach. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures SPAA*, pages 50–59, 2003.

[21] C. M. Newman and L. S. Schulman. One dimensional $1/|j-i|^s$ percolation models: The existence of a transition for $s \leq 2$. *Communications in Mathematical Physics*, 180:483–504, 1986.

[22] L. S. Schulman. Long range percolation in one dimension. *Jouranl of Physics A*, 16(17):L639–L641, 1983.

[23] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and F. D. H. Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. Technical Report TR-819, MIT LCS, 2001.

[24] D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. *Nature*, 393:440–442, June 1998.