Noname manuscript No.
(will be inserted by the editor)

Adaptive Dynamics of Realistic Small-World Networks

Olof Mogren · Oskar Sandberg · Vilhelm Verendel · Devdatt Dubhashi

April 19, 2009

Abstract Continuing in the steps of Jon Kleinberg's and others celebrated work on decentralized search, we conduct an experimental analysis of *destination sampling*, a dynamic algorithm that produces small-world networks. We find that the algorithm adapts robustly to a wide variety of situations in realistic geographic networks with synthetic test data and with real world data, even when vertices are unevenly and non-homogeneously distributed.

We investigate the same algorithm in the case where some vertices are more popular destinations for searches than others, for example obeying power-laws. We find that the algorithm adapts and adjusts the networks according to the distributions, leading to improved performance. The ability of the dynamic process to adapt and create small worlds in such diverse settings suggests a possible mechanism by which such networks appear in nature.

1 Introduction

In 1967 Stanley Milgram set out to measure the "smallness" of the world. He wanted to know if it was really true that any two people could be connected through a short chain of acquaintances. To conduct this experiment, he gave volunteers living in Omaha, Nebraska, a letter addressed to a stockbroker from outside Boston, asking them to forward it to him, with the stipulation that the letter could only ever pass between people who were on a first name basis. The results of his experiment were generally seen as proof that we really do live in a small world – for the letters that arrived successfully, the average number of steps was just six.

The idea of the small world has inspired the mathematical study of graph diameter. Roughly speaking, it has been noted that if the edges of a graph are chosen randomly, then the diameter tends to be "small": of the order of $\log n$ where n is the size of the graph. However, while such a world may be small, this does not in itself explain the success of Milgram's experiment. In his seminal paper from 2000 [14], Jon Kleinberg took an algorithmic perspective and asked: how is it that it was possible for people to know whom they should send the letter to so it would arrive in few steps? After all, the social network is a criss-crossed maze of connections, of which the participants have no overview. Kleinberg showed that for it to be possible, using only local knowledge, to efficiently forward the message to its destination the graph must have a particular form. Specifically, the probability that two people are acquainted must follow a particular power-law relation with the distance between them. When this is the case, messages can be routed in a polylogarithmic number of steps, in all other cases such paths exponentially longer. Graphs where routing is efficient (paths being polylogarithmic of the size) have since been labeled *navigable*.

O. Mogren · V. Verendel · Devdatt Dubhashi
Department of Computer Science and Engineering, Chalmers University of Technology and Göteborg University

O. Sandberg
Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA.
E-mail: ossa@google.com *This work was mostly done while this author was at Göteborg University*

1.1 Motivation

In Kleinberg's original work [14] [13], his model for the world was a two-dimensional grid, where people knew their k-nearest neighbors, and had r random long-range

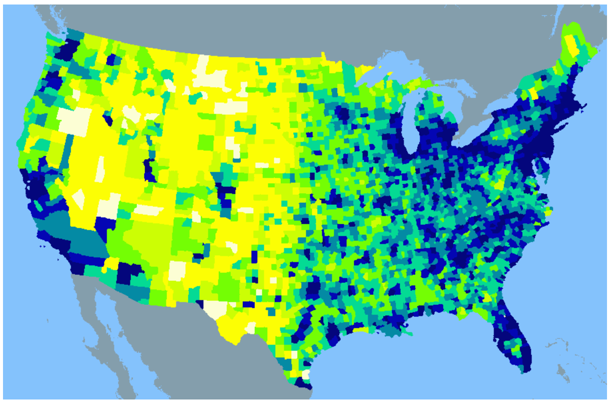

Fig. 1 In the real world, populations are in-homogeneously distributed. Here the population density of the United States of America by county.

contacts in the network. The distribution of these *shortcuts* is what determines the navigability of the graph. Later works have extended the model to more general and more realistic settings. In his PhD thesis, David Liben-Nowell [18] studied a real-world social network of people from the United States connected over the Internet. He found that this network was navigable, and could be made to fit with Kleinberg's theory, but only after adjustments had been made to take into account the highly non-homogeneous geographical distribution of the population. While his work gives hints as to in what situations the unadjusted model fails, the criteria for this have not been characterized. Several works have explored this more general relation in other contexts [15] [25] [11] [8].

Another question that is raised by any attempt to apply Kleinberg's ideas to the real world, is understanding why social networks should be navigable in the first place. In some ways, the negative results (that is, the lower bounds) in Kleinberg's work are much stronger than the positive ones: for almost all edge distributions efficient routing is not possible, it is only for distributions meeting very strict criteria that it is. This seems strange in relation to the lessons of Milgram's experiment – people really could route well – and also Liben-Nowell's observations from his dataset. It seems feasible that there is some dynamic which causes navigability to arise. Sandberg and Clarke [24] [23] have suggested such a dynamic a re-wiring algorithm which causes networks to become navigable. By simulating a large number of searches on the network, and changing the shortcuts based on the path taken by each search, the algorithm progressively creates a small-world from any starting distribution.

We consider the situation of graphs with fixed, independently chosen, edges to be largely understood: for almost any situation, there are known methods for creating navigable graphs. This paper is not an attempt to retread this ground. The goal of this paper is to see if it is possible for dynamic models, previously explored only on regular grids, to function also in networks with realistic population distributions.

To this effect, we undertake an experimental analysis of Sandberg and Clarke's algorithm under more realistic situations. As far as we are aware, this is the first comprehensive experimental analysis of a dynamic model for the emergence of small worlds in realistic geographic scenarios. We study how it behaves when vertices are not placed in a grid, but rather distributed in a continuum and with non-homogeneous population density. We contrast this with the results of using the same edge probabilities as in the homogeneous case, as well as the methods of Liben-Nowell et al. [19]. We also investigate how the algorithm responds to uneven distributions in the source and destination of searches – something more similar to the power-law ("scale-free") distributions known to be common in many real life networks. Finally, we also simulate another re-wiring algorithm, described in [6], and compare its results to those of Sandberg and Clarke.

1.2 Previous Work

For a summary of previous work in the field of navigable networks, see Jon Kleinberg's ICM survey [16]. Besides the algorithm of Sandberg and Clarke which is the main target of our studies, Clauset and Moore have suggested a different re-wiring algorithm which they find experimentally also leads to a navigable graphs. Their work remains unpublished, but a preprint is available online [6]. The two methods are superficially similar, but actually lead to very different dynamics (see Section 6 for further discussion of this method). Other models for small-world emergence have been suggested by Sandberg [22] and Duchon et al. [9] and recently by Chaintreau et al. [5] but these are not evolutionary rewiring schemes and function differently.

1.3 Contribution

We characterize our contribution as follows:

1. We investigate experimentally navigable small-world models with non-homogeneous population distributions, identifying when naively applying the method of adding shortcuts that Kleinberg used in the grid fails to produce navigability. The fact that this method

fails in some cases has been observed before [19] experimentally and analytically, but when this happens is not fully understood. Here we find that it works surprisingly well, even in cases where it is difficult to justify analytically, but we identify a family of distributions where the original model demonstrably fails to produce small worlds, while previously known density adjusted models function well.
2. We simulate the placement of shortcuts in such environments using Sandberg and Clarke's evolutionary rewiring model. We are not aware of any previous studies of dynamic small-world emergence models in realistic geographic settings. We demonstrate that the algorithm used produces navigable networks robustly under all tested circumstances: synthetic distributions with homogeneous and non-homogeneous distributions of data points, as well as scenarios based on the real world population distributions of Sweden and the United States. We also simulate creating shortcut's using Clauset and Moore's rewiring model, however these results are less conclusive.
3. We test the same evolutionary model also for non-homogeneous popularity models – when some people are more popular targets for searches than others, for instance obeying various power laws. We find that it not only works robustly in these cases, but it adapts the distribution and produces better mean results than otherwise.

The source code of our simulators and data files can be found at:

`http://www.math.chalmers.se/~ossa/dynamic/`

2 Decentralized Routing and Navigable Augmentation

Let $G = (V, E)$ belong to a family of finite graphs with high (some power of $|V|$) diameter, and let the random graph G' be created by addition (augmentation) of random edges to G. It is well known, see for instance [3], that the diameter shrinks quickly to a logarithm of $|V|$ when random edges are added between the vertices. Navigability does not concern a small diameter, however, but rather a stronger property: the possibility of finding a short path between two vertices in G' using only local knowledge at each vertex visited. By local knowledge, one means that each vertex knows distance with respect to G, but does not know which random edges have been added to any vertex until it is visited. The exact limits of such *decentralized routing algorithms* have been discussed elsewhere [14] [2], but we will only discuss the one we use: *greedy routing.*

In greedy routing for a target vertex z, the next vertex chosen is always that neighbor which is the closest to z according to the distance induced by G (with some tie-breaking rule applied). Both the original and augmented edges can be used, but because the choice is only optimal with respect to G, the path discovered by greedy routing will seldom be a minimal path in G'.

Kleinberg originally let G be a 2-dimensional $n \times n$-grid and independently added shortcuts from each vertex to random destinations. Each shortcut is added to $x \in V$ such that for $y \in V$, and some $\alpha \geq 0$

$$\mathbf{P}(x \rightsquigarrow y) = \frac{1}{h_{\alpha,n} d(x,y)^{\alpha}} \tag{1}$$

where $x \rightsquigarrow y$ is the event that x is augmented with an edge to y and $d(x, y)$ denotes L^1 distance in $\mathbb{Z}^d$. $h_{\alpha,n}$ is here a normalizing constant, equal to $\sum_{y \neq x} d(x, y)^{-\alpha}$. His observation was that when $\alpha = 2$, greedy routing between any two points in V takes $O(\log^2 n)$ steps in expectation, while for any other value of α decentralized algorithms create routes of expected length at least $\Omega(n^s)$ steps for some $s > 0$ (where s depends on α and the dimension but not the size of the graph nor the algorithm used).

One may note that for x in a 2-dimensional grid and $r > 0$, $|\{y \in V : d(x, y) \leq r\}| \propto r^2$. The general principle that may be noted by combining this with (1) is that under navigable augmentation the probability that x links to y should be inversely proportional to the number of vertices that are closer to x than y. This has been observed to hold not just when G is a grid of any dimension, but also for wider classes of graphs, see e.g. [15] [25] [8].

In particular, in Liben-Nowell et al.'s paper on geographic routing [19], they let the *rank* of a vertex y with respect to x be y's position when the vertices are ordered by distance from x, written $\mathrm{rank}_x(y)$. (Some natural ordering of the vertices is used for tie-breaking). Their augmentation principle is then that

$$\mathbf{P}(x \rightsquigarrow y) = \frac{1}{h_n \mathrm{rank}_x(y)}. \tag{2}$$

where $h_n = \sum_{k=1}^{n} k^{-1} \approx \log n$. In a companion paper by Kumar et al. [17] they prove analytically that this leads to a $O(\log^2 n)$ path lengths in expectation in a discrete non-homogeneous model (the population is confined to a two dimensional grid, but the number of people at each grid point varies).

2.1 Continuum Settings

When attempting to model reality, it is preferable to view the "world" of the vertices as a continuous metric space, rather than just a base graph G. In particular, we want both the routing and the augmentation to be with respect to the distance between vertices given arbitrary positions in the space, rather than just graph distance.

That is, if M is a metric space with $d : M \times M \mapsto \mathbb{R}$ a metric, then the set V may consist of n arbitrarily distributed points in M. (Typically, and in the text below, the metric space is a compact subset of $\mathbb{R}^2$, and d is Euclidean distance.) We then construct the "short-range" links (that is G) so as to respect the geometry of the space. In particular, one wishes for G to be suitable for greedy routing with respect to d in the sense that for $z \neq x$, x always has a neighbor closer to z than itself – if this is not the case, it is possible for a greedy route to reach a "dead-end" at which no progress can be made in the next step.

This sort of construction was considered in [7]. There, the authors let the the base graph G be constructed by connecting each vertex x with all vertices within some distance $r(n)$. For sufficiently large $r(n)$ this will with high probability lead a base graph which is suitable for greedy routing. Slivkins [25] constructs a graph purely through random augmentation with no short-range links, but with a sufficient number of augmented edges ($\Theta(n^2)$) that a base graph is with high probability never needed. Using edge probabilities similar to (2) he attains results for non-homogeneous vertex distributions, in terms of the graph size, doubling dimension, and aspect ratio (the ratio of the shortest to longest inter-vertex distance) of the model.

In [23] a different approach is used. Instead of connecting all near vertices, a Voronoi tessellation of M with respect to the points is calculated, and each vertex is connected to those with neighboring cells. Thus G is the Delaunay graph (or Delaunay triangulation) of the set of points. The advantage of this approach is that G is a planar graph more elegantly describing a neighbor structure on M, and that no probability calculations are necessary: it is easy to see that G always allows a greedy route to monotonically approach its target. Delaunay graphs can be efficiently calculated using well-known algorithms [10]. Figure 2 shows a Delaunay graph realization on a randomly chosen set of points.

Once G has been defined, one can augment it to create G' as before, adding outgoing edges to each vertex. The probabilities are found by replacing L^1 distance with the more general $d(x, y)$ in (1) and when calculating $rank_x(y)$.

2.2 Destination Sampling

"Destination sampling" is a name given to the re-wiring algorithm introduced by Sandberg and Clarke in [24].

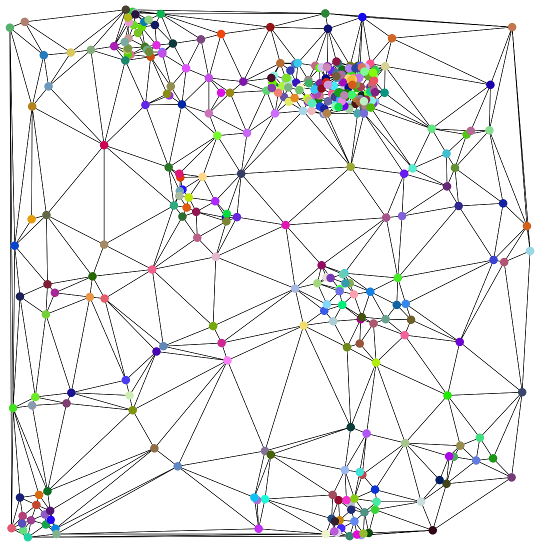

Fig. 2 Realization of 375 vertices in a *random* distribution (see Section 3.1) on $[0, 1] \times [0, 1]$ using $k = 10$ and $\gamma = 1.2$, together with the Delaunay triangulation.

This is not a method of augmenting a graph G to create G' as such, but takes any given augmentation, and changes the shortcuts (without changing their number or the out-degree of any vertex) so as to make the resulting graph navigable.

The algorithm can be expressed in varying levels of generality, but the general principle is always the same: each vertex samples the destination of its shortcuts from the destinations of searches that pass through that vertex.

Algorithm 21 *Let $G_s = (V, E \cup E_s)$ be an augmented graph at time s. $G = (V, E)$ is the base graph, and E_s the set of shortcuts, which for each vertex in V contains at least one outgoing edge.*

Let $0 < p < 1$. Then G_{s+1} is defined as follows.

1. *Choose y_{s+1} and z_{s+1} randomly from V.*
2. *If the chosen vertices are distinct, do a greedy walk in G_s from y_{s+1} to z_{s+1}. Let $x_0 = y_{s+1}, x_1, x_2, ..., x_t = z_{s+1}$ denote the points of this walk.*
3. *For each $x_0, x_1, ..., x_{t-1}$ independently and with probability p replace a randomly chosen shortcut from that vertex with one to z_{s+1}.*

See Figure 3 for an illustration of the process.

In order to create a navigable augmentation, this algorithm is applied repeatedly, causing it to converge to a stationary distribution. For simulations and analytical motivations why this works when

1. The vertices of V are homogeneously distributed.
2. y_{s+1} and z_{s+1} are chosen uniformly at random.

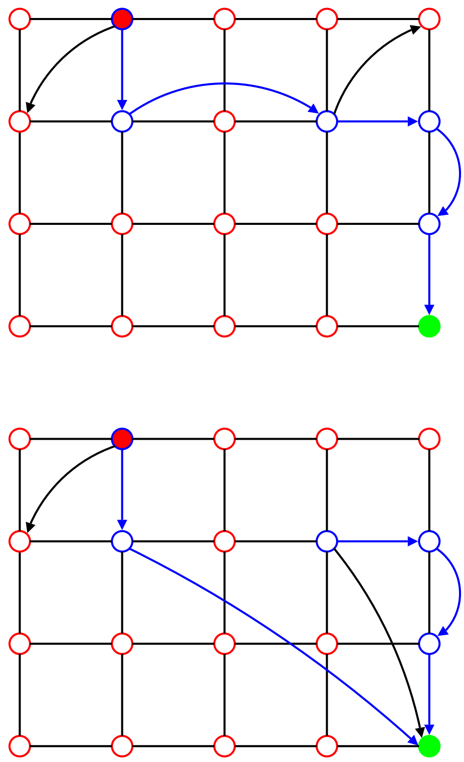

Fig. 3 An illustration of Destination Sampling on an augmented grid before and after a rewiring. The blue vertices and edges represent a greedy path from red to green. After the path has been found, the shortcuts of two vertices along the path are randomly selected to be rewired.

see [24] and [23]. The goal of this paper is to study what happens when these two assumptions do not necessarily hold, as one would expect in a realistic situation.

The parameter p in the algorithm is used to limit the dependence among edges of nearby nodes. In theory, the algorithm performs better the lower p is, but the sampling, and thus convergence, is slower. We use a value of $p = 0.1$ which we have determined experimentally provides a good trade-off, throughout the paper.

2.3 Motivating Destination Sampling

At first glance Algorithm 21 may not seem like a particularly good algorithm for explaining how social bonds are formed. Indeed, it would seem ridiculous to claim that any simple algorithm can explain the chaotic manner in which social networks are generated. We note, however, that in order to explain the presence of navigability in real world networks, it is by no means necessary for all the edges to be created in a navigable manner. Since adding extra edges to a graph can never reduce a its navigability, it is sufficient that some social bonds be created according to a pattern helpful for navigation. The presence of other friendships may sometimes help, but can never hinder, a greedy routing algorithm.

With this in mind, we do think that there is an intuition for why destination sampling bears some resemblance to real life social networks. Of course the algorithm is never formally applied in social settings – and outside psychological experiments Milgram-type social navigation almost never occurs – but the basic intuition behind Algorithm 21 is simply this: vertices (people) end up knowing those vertices that other vertices expect them to know. By sampling the edges from the destinations of the queries, the distribution of outgoing edges from a vertex is forced to be the same as the distribution of incoming queries. Replacing a "who" with a "what", this type of dynamic is almost certainly present in our everyday lives: many of the things we know, we know exactly because we have been asked them before, and then forced to find out the answer[1]. It is not a stretch to imagine that at least some social connections are formed in the same manner.

Of course, this is not a work of sociology or social psychology, and the reasoning in this section amounts to nothing more than speculation. Inventing and performing experiments to justify or contradict these statements is a very interesting further development, far outside the scope and subject of the current work.

3 Population Density

3.1 Experiments

To experiment with non-homogeneous population densities we used a continuum model and the Delaunay graph as described above. We divided a 2-dimensional real space into zones of different population density, and populated it with a non-homogeneous spatial Poisson process. The intensity of the process in each zone was the zones population density normalized so as to give approximately a desired population size for the whole space. With the vertices thus placed a Delaunay graph can then be constructed using known algorithms, and we experimented with different ways of augmenting edges to ensure navigability.

Our goals with this were twofold – firstly to identify in which types of situations augmenting according

[1] One of the authors of this paper finds himself, during its preparation, swamped with questions from his students regarding where their exam will take place. In fact, he has no better way of finding this out than the students themselves (and no personal need to know it), but having forwarded the first couple of queries to the responsible administrator, he is now in possession of the information, and can answer directly. The perception among his students that he will know the exams location, has thus caused him to learn it.

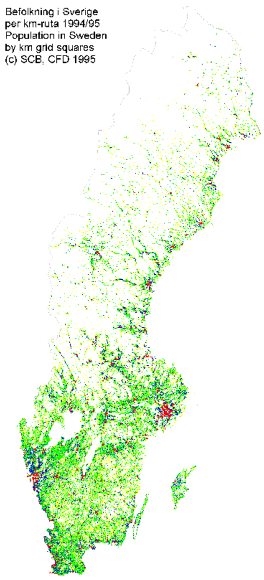

Fig. 4 The population density of Sweden, broken into 1x1 km squares.

to formula (1) fails to lead to navigability, while (2) works. Liben-Nowell et al. [19] as well as Slivkins [25] give some hints as to when this may be the case, but do not characterize it. Secondly, we simulated Destination Sampling (Algorithm 21) to see if it adapts and gives navigable augmentation even in cases where distance based augmentation does not.

The role of augmentation by rank using formula (2) is thus to provide a known "good" scenario, while distance based augmentation using formula (1) provides a known "bad" scenario. Our main goal is therefore to see whether destination sampling performs like the former, rather than the latter.

The models of population density that were used are the following:

1. *Uniform*: The n vertices are placed uniformly at random across a square space $M = [0,1] \times [0,1]$.
2. *Metropolis*: Here also the vertices are placed randomly in the same square space, but this time with 90% of the total intensity within 20% of the maximal distance from the space's center.
3. *Random*: $[0,1] \times [0,1]$ was divided into $k \times k$ equally sized square zones, which were given a randomly ordered labeling of $s = 1, \ldots, k^2$. The population of each zone was then given a relative population density of $1/s^\gamma$, making the labels an ordering from most to least densely populated. For our experiments, we used $k = 100$ and $\gamma = 1.2$ where the latter value approximates the average value of decay of city sizes in the real world [26]. See Figure 2 for an example realization.
4. *Real World*: Finally, we used data regarding the contemporary population distribution of Sweden and the United States. For Sweden, data was obtained from Statistics Sweden [27] giving the population of each of the country's 449,964 square kilometers, which we interpret as the proportional intensity of population in that area. For the United States, a map showing the population density of each county in the lower 48 states, taken from The National Atlas [21] was used[2].

In each case, each vertex was given one outgoing shortcut, selected by the following augmentation methods:

1. *Distance*: Explicit sampling according to a power-law of the distance, as in Kleinberg's original work. This means following formula (1) but with d in the formula and the normalizer interpreted as Euclidean distance in $\mathbb{R}^2$.
   For two dimensions, we use $\alpha = 2$, the value at which navigability arises in uniform networks.
2. *Rank*: Explicit sampling, but using the rank formula (2) as used by Liben-Nowell et al. This is largely equivalent to other "inverse ball" type augmentation methods, as used in e.g. [15] [25] [8].
3. *Destination Sampling*: Each node is initiated with no useful long-range link (formally, it has one to itself), and then the Algorithm 21 is run $10n$, where n is the graph size, times.

In some cases, we also compared with the results of choosing the shortcut uniformly among the other vertices. This is known to give greedy path lengths which are a fractional power of the number of vertices – $\Omega(n^{1/3})$ with a uniform population distribution – and thus was used only as a baseline for comparison.

We note that the number of iterations of Algorithm 21 we perform, $10n$ is very small compared to the number of vertex pairs that can be routed between. We have no theoretical motivation for using so few iterations, and can refer only to the simple fact, as observed below, that it is sufficient to create navigable networks. We consider its remarkably fast convergence rate to be a strength of the destination sampling method. In fact, simulations indicate that even $2n$ iterations is enough in all cases (see Figure 6) – we simulate more simply because compared with the other augmentation methods tested the computation time needed for destination sampling is insignificant.

[2] The data in both cases was not exact. The map of Sweden gives the population of each square kilometer among the levels 0, 1-4, 5-29, 30-149, 150-4999, and 5000 and above. The map of the continental USA was divided in to 0, 1-4, 5-9, 10-24, 25-49, 50-99, 100-249, and 250 and above people per square mile.

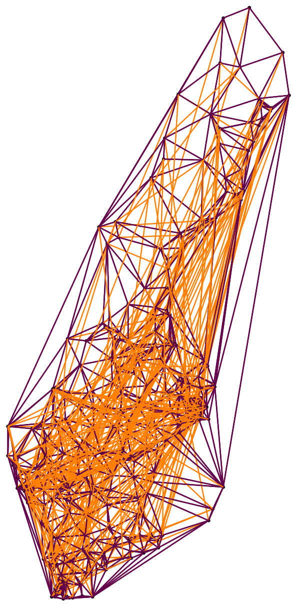

Fig. 5 A realization of a Delaunay graph and destination sampling on a population distributed according to Sweden's population density (see Figure 4).

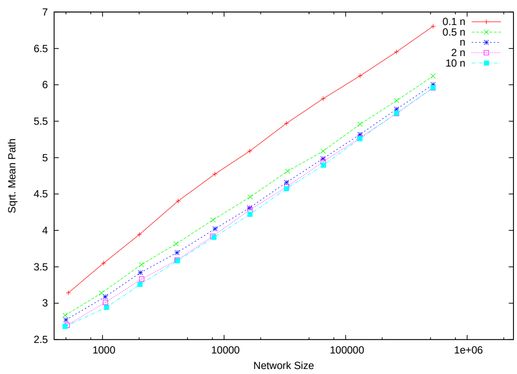

Fig. 6 The performance of greedy routing after destination sampling for different numbers of iterations. We are not able to see any detectable improvement in performance after $2n$ iterations.

3.2 Results

Our results on non-homogeneous population distributions are shown in Figures 7 – 12. In general, we find that adding shortcuts as done by Kleinberg (1) in the grid can work well even when the population is not uniformly distributed. This is shown by the fact that in the *metropolis* model, where most of the population is limited to a central core, we still get $\log^2 n$ scaling of the path lengths, see Figure 8. We find that in order for the purely distance based augmentation to fail, we have to turn to highly irregular population models. Previously it was known (see Liben-Nowell et al. [19]) that the distance based augmentation could

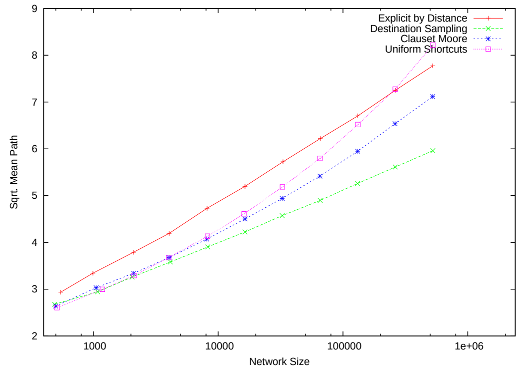

Fig. 7 Performance of greedy routing when augmenting the Delaunay graph of uniformly randomly distributed points in $[0,1] \times [0,1]$, using distance based augmentation (1) with $\alpha = 2$, rank based augmentation (2), destination sampling, and also Clauset and Moore's algorithm (see Section 6).

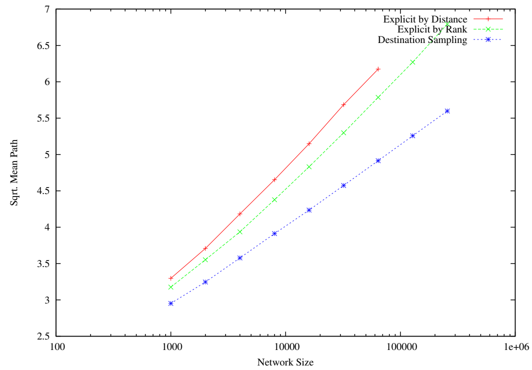

Fig. 8 Performance of greedy routing when augmenting the Delaunay graph of *Metropolis* distributed points in $[0,1] \times [0,1]$, using distance based augmentation augmentation (1) with $\alpha = 2$, rank based augmentation (2), and destination sampling.

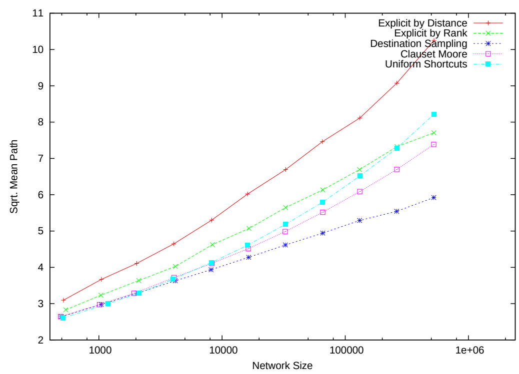

Fig. 9 Performance of greedy routing when augmenting the Delaunay graph of *Random* model distributed points in $[0,1] \times [0,1]$, using distance based augmentation augmentation (1) with $\alpha = 2$, rank based augmentation (2), destination sampling, choosing shortcuts uniformly, and also Clauset and Moore's algorithm (see Section 6).

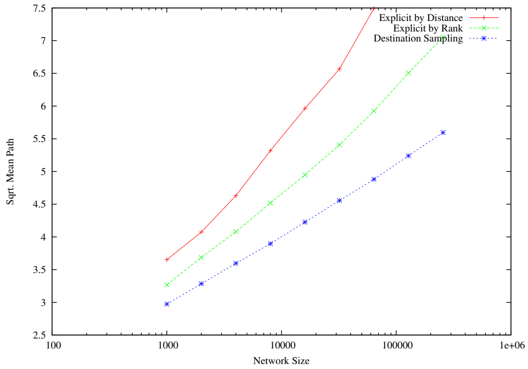

Fig. 10 Performance of greedy routing when augmenting the Delaunay graph of points distributed according to Sweden's population, using distance based augmentation (1) with $\alpha = 2$, rank based augmentation (2), and destination sampling.

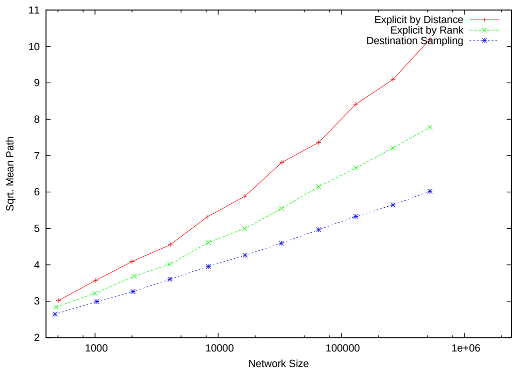

Fig. 11 Performance of greedy routing when augmenting the Delaunay graph of points distributed according to the population density of the United States, using distance based augmentation (1) with $\alpha = 2$, rank based augmentation (2), and destination sampling.

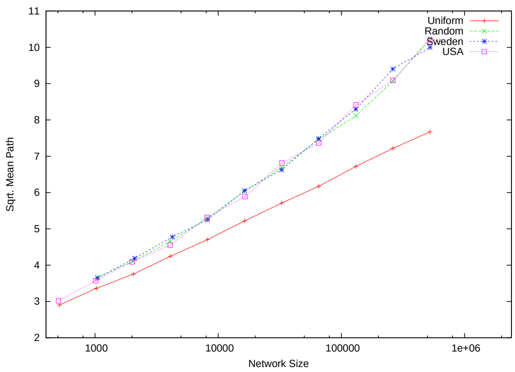

Fig. 12 Explicit distance based augmentation gives the same (poor) performance in the Sweden, USA, and *Random* population models.

fail, but it is not exactly clear under what circumstances. We identify one such family, of irregular non–homogeneous distributions, namely the *random* model described above, where distance based augmentation performs worse than even uniform such (Figure 9).

Figures 10 and 11 show the results using the real world data of the population density in Sweden and the USA. As expected due to Liben-Nowell et al.'s observations of the Internet community data, as well as our structurally similar *random* configuration, purely distance based augmentation does not give good result here, being beaten even by uniform augmentation.

Remarkably, the results for the *random* model are almost identical to those for the real world populations (Figure 12), showing that as far as distance based augmentation is concerned, the real world data appears structurally very similar to that produced by the random model.

The destination sampling algorithm performs well in *all* situations – both synthetic test data as shown in Figures 7 - 12 and for real world data, as shown in Figures 10 and 11 – always producing results that scale like $\log^2 n$ as desired, and consistently performing better than explicit distance or rank based augmentation. We have not been able to find *any* population distribution or situation where the re-wiring algorithm performs poorly.

4 Popularity Distributions

4.1 Experiments

One of the most striking differences between social networks and most simple random graph models is that the former seem to have power-law degree distributions, while the latter most often have Poisson distributed or even constant degrees. The celebrated "preferential attachment" model (see [1] and [20] as well as [4] for rigorous analysis) explains this fact by showing that such distributions arise when new vertices are more likely to connect to vertices which already have a high degree.

In the context of the re-wiring algorithm, one might expect some vertices to be more popular targets of searches than others (as would most definitely be the case if the algorithm was, for example, used to wire up a peer-to-peer network). It is of interest to see whether the rewiring algorithm can adapt also to this situation. (Clearly the distribution of searches has a large effect on what edges are formed during destination sampling.)

In order to separate these results from those above, we return to the original model of Kleinberg – vertices placed in a regular lattice with connections to their

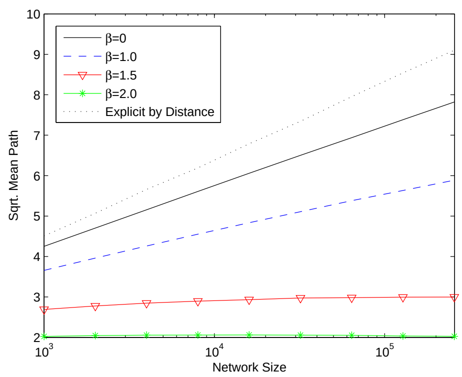

Fig. 13 Performance of greedy routing when choosing destination by power-law distributions of different exponent (β in (3)), using destination sampling in a one dimensional grid. This is contrasted against explicit augmentation by distance (with $\alpha = 1$), where the destination distribution of course makes no difference.

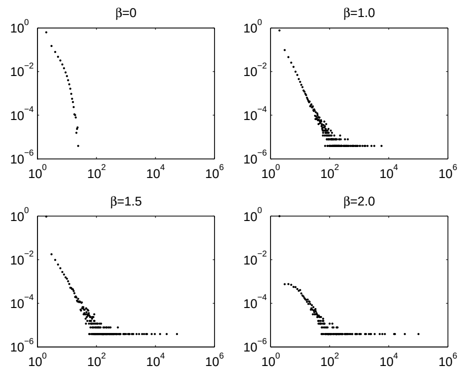

Fig. 14 Degree distributions of the graphs created by destination sampling when choosing choosing destinations by power-law, for different values of the exponent (β) in (3). The network has 256,000 vertices, each with out-degree one. The plots show the fraction of vertices with each total in-degree (rounded up to the nearest multiple of ten).

nearest neighbors, as well as a single outgoing long-range contact. The out-degree of each vertex is thus still fixed, but the in-degree varies as a result of target popularity.

In our experiments, we produce a random order of the vertices, and consider each vertex x's position in this order, $p(x)$, as its popularity ranking. We then select the targets of queries using power-laws, of the form

$$\mathbf{P}(\text{choosing } x \text{ as a target}) \propto p(x)^{-\beta} \tag{3}$$

for β ranging from 0 to 2. As well as evaluating the performance of the destination sampling under these conditions, we also study the resulting degree distributions to see if a power-law is actually recovered.

4.2 Results

Figures 13 – 14 show our results when we return to a homogeneous grid, but instead let the popularity of vertices as destinations vary. As in the cases above, we find the destination sampling excels, producing shorter paths than explicit augmentation. In fact, as β increases in (3), we find that destination sampling gives shorter and shorter paths. Intuitively, this follows from the fact that since most searches are going to a limited set of vertices, most shortcuts also lead to those vertices, allowing most of the routing to occur within a small subset of the whole graph. This is most clear when $\beta > 1$, in which case one can for any $\epsilon > 0$ fix an m such that at least $1 - \epsilon$ of the queries are destined for the m most popular vertices, independent of the graph size n. Indeed, one can see in Figure 13 that we observe no increase in the path lengths for $\beta = 2$ after a certain point (it stays below 2.1 mean steps for all sizes). In contrast, both the distance based augmentation as well as rank based augmentation, which assign fixed probabilities independently of the popularity distribution, of course do not take advantage of the non-uniform popularity distribution at all.

The resulting degree distributions are described in Figure 14. One can see that when the destination is selected according to a power-law distribution, the degree of the vertices also end up following such a distribution. This is not particularly surprising, given the way the algorithm functions, but shows that we can generate power-law ("scale-free") graphs without sacrificing navigability.

5 Combined

For completeness, we look at what happens when performing destination sampling on the Swedish population model from the first section, while at the same time using a biased popularity distribution as in the second.

As expected from the above results, destination sampling functions well also when combining both a non-homogeneous geographical population density, and a power-law distribution of destination, see Figure 15. The mean path length is considerably shorter than that attained under either the distance or rank based augmentations.

6 Clauset and Moore's rewiring scheme

To round off our experiments, we attempt a comparison of our results using destination sampling with those of

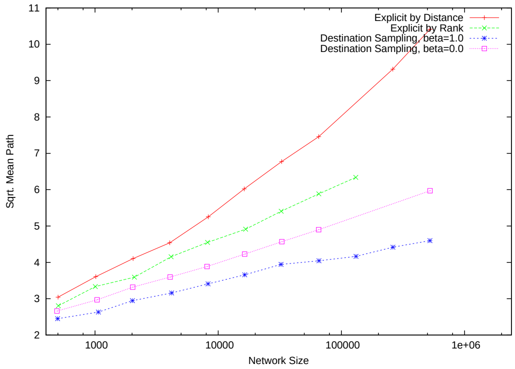

Fig. 15 Performance of greedy routing when combining the *Sweden* population distribution with a power-law popularity of the destinations (β as in (3)).

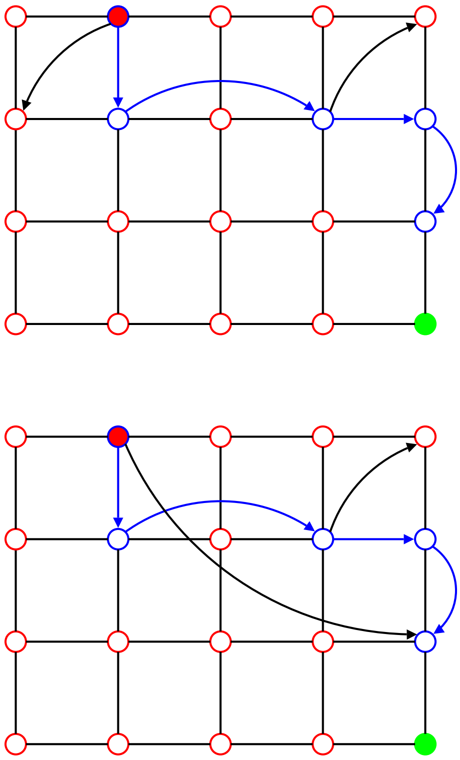

Fig. 16 An illustration of Clauset and Moore's algorithm on them same augmented grid as in Figure 3 before and after a rewiring, with $\tau = 4$. The blue vertices and edges represent a greedy path from red towards green. Because green has been reached after τ steps, red's shortcut is rewired to point to where the path ended.

a different rewiring algorithm described by Clauset and Moore [6]. Like Algorithm 21, the CM algorithm starts from a graph with only self-loops as shortcuts, and simulates a large number of queries. Using the same definitions as in 21, the main steps of their algorithm are as follows:

1. Choose y_{s+1} and z_{s+1} randomly from V.

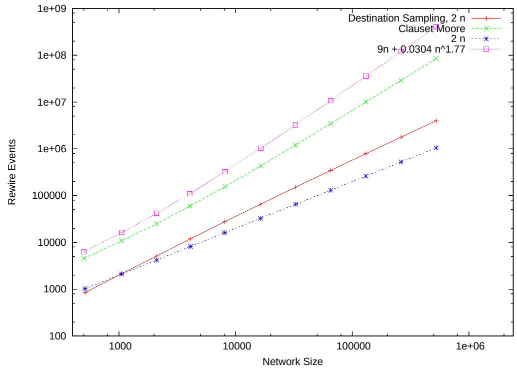

Fig. 17 The number of rewire events needed for destination sampling and Clauset and Moore's algorithm to reach optimality. Plotted also are the lines for $2n$ and $9n + 0.0304n^{1.77}$: the number of paths simulated respectively. Note that both axis are logarithmic.

2. Let $d = d_G(y_{s+1}, z_{s+1})$ be the distance from y_{s+1} to z_{s+1} in the base graph, and select τ uniformly from $1, 2, \ldots, d$.
3. If the chosen vertices are distinct, do a greedy walk in G_s from y_{s+1} to z_{s+1} stopping when z_{s+1} is reached, or when τ steps have been taken (whichever comes first). Let $x_0 = y_{s+1}, x_1, x_2, ..., x_t$ denote the points of this walk.
4. If $t = \tau$, replace one of y_{s+1}'s shortcuts with one to x_t.

This algorithm is equivalent to "giving up" routing if the destination has not been reached within τ steps, and then attempting to shorten the route by rewiring the starting vertex's shortcut to point to the vertex at which routing stopped. See Figure 16 for an illustration. In [6], the authors show experimental results indicating that this algorithm does lead to a navigable graph when using a one dimensional base graph and distance function. They estimate, based on simulations, that about

$$9n + 0.0304n^{1.77}$$

requests are needed to reach the optimum path length. This is considerably more than the n to $2n$ iterations which we estimate are needed for destination sampling (see Figure 6). Part of this can be explained by the fact that destination sampling rewires several vertices per request, while Clauset and Moore's algorithm rewires at most one (and in practice much fewer), but destination sampling is considerably faster also in terms of rewire events (see Figure 17).

Compared to those in [6], our results for 2-dimensional continuous settings are less conclusive. The results of simulations according to CM have been included in Figures 7 (with *Uniform* population density) and 9 (with the *Random* density model). These two models were

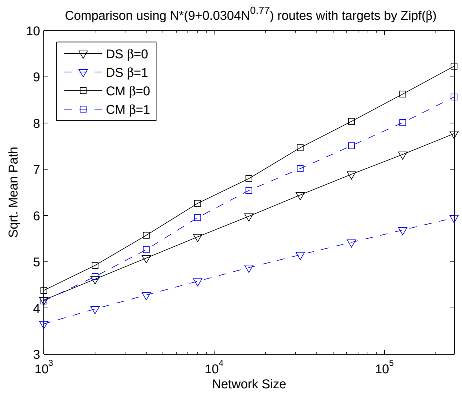

Fig. 18 Comparison of Destination Sampling with Clauset and Moore's algorithm with biased popularity distributions, as in Section 4.

chosen because they are representative of the different behaviors observed in the simulations above. In order to make sure that convergence times are not an issue, we simulated twice as many iterations for these experiments as [6] indicates is necessary[3]. In absolute terms, the results look good, with CM beating the distance and rank based augmentations for all network sizes. The performance is not degraded when moving from the uniform to the random population distribution, indicating that CM, like destination sampling, adapts. The plotted lines are not unambiguously linear however, and so it is unclear what asymptotic trend is indicated. Whether or not we are seeing true navigability emerge with CM is unclear, and even after considerable effort we are forced to leave the question unanswered.

We also simulated comparisons of destination sampling and CM in the popularity distribution models discussed in Section 4. Here CM leads to greedy routes of comparable length to destination sampling, although it still requires many more simulated requests for convergence. See Figure 18.

7 Conclusion

We find, as has been observed before, that in geographic networks the augmentation process is sensitive to the environment and distribution of the vertices. If the distribution deviates sufficiently in structure from a uniform placement of the vertices, otherwise effective methods of assigning shortcuts will not work in a number of cases. The destination sampling algorithm, however, is adaptive and will create navigable graphs in all cases that we have studied.

Likewise, when the popularity of the different vertices as destinations for searches is uneven, the destination sampling algorithm will adapt and is able to utilize this to find even shorter paths.

Given its remarkable ability to create augmentations reflective of each situation, we believe that in any case where navigable augmentation is possible (see [8] [12] for a discussion on the limits of this) destination sampling can be used to achieve navigability. We also note that the formulation of the algorithm requires no understanding of the actual situation - the same exact procedure will work regardless of geographic or popularity distribution, which is not true, for instance, when augmenting by rank.

[3] Doing more iterations than this, or simulating even larger sizes, becomes difficult without parallelization, since our largest simulations already require around 2^{36} routing operations. Parallelization is clearly possible, but it is hard to estimate what effect this will have on the convergence rate.

References

1. A.L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509512, 1999.
2. L. Barriere, P. Fraigniaud, E. Kranakis, and D. Krizanc. Efficient routing in networks with long range contacts. In *Proceedings of the 15th International Symposium on Distributed Computing, DISC'01*, pages 270–284, Berlin / Heidelberg, 2001. Springer.
3. B. Bollobás and F. Chung. The diameter of a cycle plus a random matching. *SIAM Journal on Discrete Mathematics*, 1:328–333, 1988.
4. B. Bollobás and O. Riordan. The diameter of a scale-free random graph. *Combinatorica*, 24:5, 2004.
5. A. Chaintreau, P. Fraigniaud, and E. Lebhar. Networks become navigable as nodes move and forget. Preprint, 2008, arXiv:0803.0248.
6. A. Clauset and C. Moore. How do networks become navigable? Preprint, 2003, arXiv:cond-mat/0309415.
7. M. Draief and A. Ganesh. Efficient routing in Poisson small-world networks. *Journal of Applied Probability.*, 43:678–686, 2006.
8. P. Duchon, N. Hanusse, E. Lebhar, and N. Schabanel. Could any graph be turned into a small world? *Theoretical Computer Science*, 355:96 – 103, 2006.
9. P. Duchon, N. Hanusse, E. Lebhar, and N. Schabanel. Towards small world emergence. In *SPAA '06: Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures*, pages 225–232, New York, NY, USA, 2006. ACM.
10. S. Fortune. A sweepline algorithm for Voronoi diagrams. In *Proceedings of the second annual symposium on Computational geometry*, pages 313–322, Yorktown Heights, New York, 1986.
11. P. Fraigniaud. Greedy routing in tree-decomposed graphs: a new perspective on the small-world phenomenon. In *Proceedings of the 13th European Symposium on Algorithms (ESA)*, pages 791–802, 2005.
12. P. Fraigniaud, E. Lebhar, and Z. Lotker. A doubling dimension threshold theta(loglog n) for augmented graph navigability. In *Proceedings of the 14th European Symposium on Algorithms (ESA)*, 2006.
13. J. Kleinberg. Navigation in a small world. *Nature*, page 845, 2000.

14. J. Kleinberg. The small-world phenomenon: an algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC)*, 2000.
15. J. Kleinberg. Small-world phenomena and the dynamics of information. In *Advances in Neural Information Processing Systems (NIPS) 14*, pages 431–438, Cambridge, Massachussets, 2001. MIT Press.
16. J. Kleinberg. Complex networks and decentralized search algorithms. In *Proceedings of the International Congress of Mathematicians (ICM)*, 2006.
17. R. Kumar, D. Liben-Nowell, J. Novak, P. Raghavan, and A. Tomkins. Theoretical analysis of geographic routing in social networks. Technical Report MIT-CSAIL-TR-2005-040, Computer Science and Artificial Intelligence Labratory, MIT, 2005.
18. D. Liben-Nowell. *An Algorithmic Approach to Social Networks*. PhD thesis, MIT Computer Science and Artificial Intelligence Laboratory, 2005.
19. D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geograph routing in social networks. In *Proceedings of the National Academy of Science*, volume 102, pages 11623–11628, 2005.
20. M. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.
21. United States Department of the Interior. National atlas of the united states. http://nationalatlas.gov/.
22. O. Sandberg. Double clustering and graph navigability. Preprint, 2007, arXiv:0709.0511v1.
23. O. Sandberg. Neighbor selection and hitting probability in small-world graphs. To appear in *The Annals of Applied Probability*, 2007, arXiv:math.PR/0702325.
24. O. Sandberg and I. Clarke. The evolution of navigable small-world networks. Technical Report 2007:14, Department of Computer Science and Engineering, Chalmers University of Technology, 2007, arXiv:cs.DS/0607025.
25. A. Slivkins. Distance estimation and object location via rings of neighbors. In *Proceedings of the Annual Symposium on the Principles of Distributed Computing (PODC)*, volume 24, pages 41–50, 2004.
26. Kwok Tong Soo. Zipf's law for cities: A cross country investigation. *Regional Science and Urban Economics*, 35:239–263, 2005.
27. Statistiska Centralbyrån (Statistics Sweden). Rutkarta över sveriges befolkning, 1995. http://www.scb.se/templates/Standard____20278.asp.