Towards small world emergence

Philippe Duchon[1], Nicolas Hanusse[1], Emmanuelle Lebhar[2] and Nicolas Schabanel[2]

27 février 2006

Résumé

We investigate the problem of efficiently preprocessing a network, in a fully decentralized manner, so that the resulting network is a navigable small world, i.e. in which the greedy routing algorithm computes paths of polylogarithmic expected length between any pair of nodes. Previous small world augmentation processes require global knowledge of a network and centralized computation, which is unrealistic for large decentralized networks. Our algorithm, based on a careful sampling of a set of leader nodes, bypasses these limitations.

1 Introduction

In this paper, we investigate the problem of efficiently preprocessing a large network, in a fully distributed manner, so that the resulting network is a navigable small world. Namely, by adding a single entry to each address book, we obtain a network in which greedy routing computes paths of polylogarithmic expected length between any pair of nodes. This problem arises as an application of recent investigations on the small world phenomenon in real interaction networks (e.g. social networks, or peer-to-peer networks). This phenomenon consists in the combination of a low diameter *and* the ability, for each node, to discover short paths without the global knowledge of all connections, as exhibited in the Milgram's seminal experiment [**?**].

The first graph model reproducing the small world navigability was proposed by Kleinberg in 2000 [**?**] and consists in a 2-dimensional regular grid augmented by a constant number of random links per node, distributed according to the 2-harmonic distribution. Kleinberg shows that greedy routing, which chooses at each step the closest node to the target among its neighbors according to the globally known grid distance, computes paths of polylogarithmic expected length between any two nodes. Further investigations point out the general characteristics of these models, by extending it to d-dimensional tori [**?**], suggesting a more general model. Recently, several augmentation processes have been proposed for larger graph classes [**?**, **?**, **?**], respectively bounded treewidth, bounded doubling dimension, and bounded growth graphs ; results are summarized in Table 1. These classes are often considered as reasonable approximations of real

Ref.	Underlying structure	Out-degree	Expected path length	Scheme
[?]	Treewidth k	1	$O(k \log k \log^2 n)$	Centralized description
[?]	Group structure[1]	$O(\log^2 n)$	$O(\log^2 n)$	
[?]	α doubling dimension[2]	$O(2^{O(\alpha)} \log n \log \Delta)$	$O(\log n)$	
[?]	b-moderate growth[3]	1	$O((\log n)^{5/2+2b})$	
this paper	$O(1)$ expansion rate	1	$O(\log^2 n)$	Decentralized

TAB. 1 – Small world augmentation processes.

networks and represent *navigable small-worlds*. The key to these results is to create random shortcuts at all distance scales. However, these later processes are essentially centralized (in particular, they require the global knowledge of the network) and therefore could hardly be implemented in the context of large spontaneous networks. Moreover, this does not provide a convincing explanation for the omnipresence of the small world phenomenon observed in interaction networks.

Our contribution. In this paper, we present the first distributed algorithm to augment arbitrary bounded growth graphs into small worlds. Our process uses a sampling step to construct a relevant approximation of the network structure. We consider the following framework. We are given a graph G representing a virtual network with a distance labeling scheme (i.e., where each node has a unique ID such that one can compute the hop-distance in G between any two nodes from their IDs). The aim is to add one single random arc per node in the graph (i.e. one single entry to each routing table in the virtual network), such that a greedy routing using the preexisting distance labeling computes a route of polylogarithmic expected length between any pair of nodes. Efficient distance labeling exist for large classes of graphs, including the ones we consider (see [?, ?, ?, ?, ?]) ; but these are often expensive to compute. Our scheme allows to improve the overall routing performance of the network without recomputing its distance labeling.

More precisely, our scheme first constructs a random tree-shaped overlay network whose subtrees correctly approximate the graph metric. Then, this tree is used to compute and route the requests for random shortcuts at all distance scales. Our algorithm is fully distributed and asynchronous, and, with high probability, requires $O(\log n \log D)$ memory per node, $O(n \log n \log D)$ messages of size $O(\log n)$, and $O(n)$ time. This paper improves in particular the results presented in [?] : it avoids complete flooding by computing in polylogarithmic time a good estimation of the graph metric.

[1] A group structure is a framework which includes metrics of polynomial ball growth and hierarchies.

[2] The doubling dimension of a metric is α if each ball of radius $2r$ can be covered by 2^α balls of radius r. δ is the aspect ratio of the metric,*i.e.* the ratio of the largest distance over the smallest one.

[3] A graph has b-moderate growth if the ratio of a ball of radius $2r$ is at most $O(\log^b r)$

Theorem 1 *Any c-bounded* [1] *n-nodes graph of diameter D can be turned into a navigable small-world graph with $O(n \log n \log D)$ messages on expectation, $O(\log n \log D)$ space of memory per node w.h.p. and $O(n)$ rounds.*

Related works. Successful known small world augmentation processes on bounded growth graphs [**?**, **?**] rely on pairing each node to a random node at distance r with probability $1/b_u(r)$, where $b_u(r)$ is the number of nodes at distance $\leq r$ in G from u.

Principle of our algorithm. In order to avoid expensive exploration of the graph, we build a tree-shaped overlay network which parsimoniously encodes good approximation of the sizes of the balls centered on each node with exponentially increasing radii (which is enough for our purpose). Indeed, thanks to the bounded growth, in order to draw its random shortcut, each node uses the ball of radius 2^i of its ancestor at level i in the tree, as a good approximation its own ball of radius 2^i. Each node of level i approximates its ball by the union of the subtrees of its sons of level $i-1$ at distance $\leq 3.2^i$. Each node starts at level 0 and increases its level with some probability or stops if level $\log D$ is reached. The subtrees are disjoint and the probability to join the upper level is suitably adjusted so that the tree covers completely the graph with high probability and thus the approximated balls are correct up to a constant factor (which is enough for our purpose).

2 Preliminaries

2.1 Notation et model

For every node u, we assign a long-range contact L_u, designed by the small-worldization process. $B_u(r)$ is the set of nodes at distance at most r from u within graph G. $b_u(r)$ is the cardinality of $B_u(r)$.

2.2 Probabilistic properties of bounded growth graph

Technical probabilistic Lemmas 2 and 3 guarantee that the random construction executed in Algorithm 3.1 succeeds with high probability, namely, they guarantee that every ball of radius 2^i contains at least one node of level i (the cover property).

Lemma 2 *Let $G = (V, E)$ be a c-bounded growth graph with total number of vertices N. Assume some random set of vertices S is selected by including each vertex independently, and such that u is included with probability p_u at least $\min(1, C \ln(N)/b_u(r))$ and at most $C' \ln(N)/b_u(r)$, where C and C' are arbitrary constants such that $C \geq 8c$.*

times the size of the ball of radius r and of same center, and the size of a sphere of radius r is at most $1/r$ times the size of the ball of same center and same radius.

[1] A graph is c-bounded if for any node u, $|B_u(2r)|/|B_u(r)| \leq c$.

Then, for any positive integer k, with probability $1 - O(1/N^2)$, each ball with radius $(2^k - 1)r$ in G contains at least one element of S, and at most $2c^k C' \ln(N)$.

Proof. We only prove the lower bound for $k = 1$; if "small" balls contain elements of S, then larger balls do to. We can assume that $b_u(r) > C \ln(N)$ for all u, as any u with $p_u = 1$ will only result in balls that are certain to intersect S.

Let u be some arbitrary vertex, and set $B = B_u(r)$. For any $v \in B$, $B_v(r) \subset B_u(2r)$, so that $b_v(r) \leq c b_u(r)$. Also, $B \subset B_v(2r)$, so that $b_v(2r) \geq b_u(r)$ and $b_v(r) \geq b_u(r)/c$. Thus, for the event $\mathcal{E}_v = \{v \in S\}$, we have

$$\frac{C \ln(N)}{c b_u(r)} \leq \mathbf{Pr}(\mathcal{E}_v) \leq \frac{cC' \ln(N)}{b_u(r)} \tag{1}$$

Thus, the probability that S contains *no* vertices in $b_u(r)$, is at most

$$\left(1 - \frac{C \ln(N)}{c b_u(r)}\right)^{b_u(r)} \leq e^{-(C/c)\ln(N)} = N^{-C/c}.$$

In other words, choosing $C \geq 3c$ ensures that this probability is smaller than N^{-3}; summing this bound over all N possible centers, the probability that at least one such ball contains no vertices in S is no larger than N^{-2}.

We now prove the upper bound, first for $k = 1$, then for general k. For any $v \in B_u(r)$,

$$\frac{C}{c}\frac{\ln(N)}{b_u(r)} \leq p_v \leq cC'\frac{\ln(N)}{b_u(r)}.$$

Thus, the number of vertices in $S \cap B_u(r)$ is the sum of independent Bernoulli trials with total expectation at least $C \ln(N)/c$ and at most $cC' \ln(N)$. Through the usual Chernoff bound (see, e.g., Theorem 4.1 in [**?**] for the version we are using here), the probability that it is higher than $2cC' \ln(N)$ (that is, at least twice its expectation) is at most

$$\left(\frac{e}{4}\right)^{(C/c)\ln(N)} = N^{-(2\ln(2)-1)C/c}.$$

Selecting $C \geq 8c$ ensures that this probability is smaller than N^{-3}, and the union bound again ensures that the probability that even one such ball contains more than $2cC' \ln(N)$ is at most N^{-2}.

For larger k, the lower bound on the expected size of $S \cap B_u(r)$ is of course a lower bound on the expected size of $S \cap B_u((2^k - 1)r)$. For the upper bound, we use the fact that, if $v \in B_u((2^k - 1)r)$, then $B_v(r) \subset B_u(2^k r)$, so that we have

$$\frac{1}{c^k} b_u(r) \leq b_v(r) \leq c^k b_u(r).$$

Plugging the lower bound part of this inequality into the expression for the p_v, and summing over all $v \in B_u((2^k - 1)r)$, we get that the expected size of

$S \cap B_u((2^k-1)r)$ is at most $c^k C' \ln(N)$, and the rest of the proof is similar to the case $k = 1$. □

In our proofs, we will make use of a slightly different version of Lemma 2, where the events $(\mathcal{E}_v)_{v\in V}$ are not independent.

Lemma 3 *Assume graph G and constants C and C' are as in Lemma 2, and that we have a collection $(X_u)_{u\in V}$ of independent, uniform $[0,1]$ random variables, and a set of integers $(\tilde{b}_u)_{u\in V}$, such that, for each u,*

$$\frac{b_u(r)}{C'} \leq \tilde{b}_u \leq \frac{b_u(r)}{C}.$$

Then, for the set $S \subset V$ defined by

$$S = \{v \in V : X_v \leq \ln(N)/\tilde{b}_v\},$$

the conclusions of Lemma 2 hold.

Proof. Consider the two sets S_0 and S_1, which both satisfy the conditions of Lemma 2 :

$$\begin{aligned} S_0 &= \left\{v \in V : X_v \leq \frac{C\ln(N)}{b_u(r)}\right\} \\ S_1 &= \left\{v \in V : X_v \leq \frac{C'\ln(N)}{b_u(r)}\right\} \end{aligned}$$

The conditions on $\tilde{b}_u$ imply that $S_0 \subset S \subset S_1$ holds with probability 1. The lower bound of Lemma 2 (on S_0) implies the lower bound for S, and the upper bound (on S_1) implies the upper bound for S. □

Lemma 4 *If an α-approximation of balls size can be computed, the cover property holds with high probability.*

Proof. Taking $C = ...$ and $C' = ...$ □

3 The overlay network

3.1 Overlay network construction

Overlay network tree-like structure : The overlay network is composed of a tree of height $\log D$ and a sequence of graphs $G_0, \ldots, G_i, \ldots, G_{\log D}$ connecting nodes of same level. For all i, two nodes of level i are connected in G_i if their distance in G is $\leq 2^i$. For each node, ℓ_u denotes the highest level reached by u. u has one self-copy in the overlay network for each level in $0, \ldots, \ell_u$, each of these copies are connected to further and further neighbors in the graphs G_i.

Overlay network at a node u : For each level $i \leq \ell_u$, u stores :

– a *parent* : one node v of level $i+1$, noted P_u^i, such that $d_G(u,v) \leq 2^{i+1}$ ($P_u^i = u$ if $i < \ell_u$) ;
– its *neighbors* : the list of nodes $N_u^i = \{w : \ell_w \geq i \text{ and } d_G(u,w_j) \leq 5.2^i\}$ *and* the size of the subtrees T_w^i rooted at each $w \in N_u^i$. Note that u belongs to N_u^i whenever $i \leq l_u$. For convenience, we distinguish the *close neighbors* $\tilde{N}_u^i = \{w \in N_u^i : d_G(u,w) \leq 3.2^i\}$.
– its *children* : the list of nodes $C_u^i = \{v : P_v^{i-1} = u\}$.

For each $i \leq \ell_u$, we define the tree T_u^i rooted on the copy at level i of u whose subtrees are the T_v^{i-1}, for $v \in C_u^i$.

The function INFORM(u,i) consists in flooding a message from node u to all the nodes at distance at most 3 from u in G_i using a BFS. u waits for an acknowledgment at the end of this flooding, required for u to decide whether it stays at level i.

Algorithm 1 Contruction of the overlay network

INPUT : a c-bounded growth graph, c and $\ln n$ given
Set $C' = 8c^4$
Choose $X_u \in [0,1]$ uniformly at random ;
$T_u^0 = \{u\}$, $i = 0$, $N_u^0 \leftarrow$ the set of nodes at distance 5 from u (BFS exploration).
$\tilde{b}_u(1) = b_u(1)/C'$
while $X_u < \frac{\ln n}{\tilde{b}_u(2^i)}$ and $B_u(2^i) \neq N_u^i$ **do**
  $i \leftarrow i+1$.
  INFORM(u,i) that u is now at level i
  Wait until all the nodes at distance at most 3 hops in G_{i-1} send their decision of staying (or not) at a same level in order to build N_u^i.
  $\tilde{b}_u(2^{i+1}) = \frac{1}{C'} \sum_{v \in \tilde{N}_u^i} |T_v^i|$;
$\ell_u \leftarrow i$.

Upon reception of message Inform$(v, i+1)$ from a neighbor $v \in N_u^i$ **:**
if u stays at level $\ell_u = i$ and $d(v,u) \leq 2^{i+1}$ **then**
  Choose v as a parent and send "$P_u = v$" to v.

Upon reception of message $P_v = u$ **from a neighbor** $v \in N_u^{i-1}$ **:** add v to the list of children C_u^i.

3.2 Overlay network properties

During the hierarchical sampling process, we will use Lemma 3 repeatedly to prove that, with high probability, every node u of level i or higher will be able to compute its neighbors $N_u^{(i)}$ and ball size estimates $\tilde{b}_u(2^i)$.

3.2.1 Connectivity of G^i

The next Lemma ensures the correctness of Algorithm 1 in terms of incremental construction of graphs G^i, namely, each node u is able to compute N_u^i,

for any i.

Consider two nodes u and v in G_i (that is, both u and v have level i or higher).

Lemma 5 *With high probability, G_i contains a path of length at most $1 + d_G(u,v)/2^{i+1}$ between u and v.*

Proof. Consider a shortest path (in G) $u = u_0, u_1, \ldots, u_\ell = v$ from u to v, and nodes $w_j = u_{(2j-1)2^i}$ ($1 \leq j \leq \ell/2^{i+1}$), regularly spaced on the path. With high probability, each ball $B_{w_j}(2^i)$ contains at least one node s_j with level at least i. Triangle inequalities imply that s_j and s_{j+1} are within distance 4.2^i of each other, and are thus neighbours in G_i. Similarly, u and s_1 are neighbours, and so are $s_{\ell/2^{i+1}}$ and v, so that the required path is $(u, s_1, \ldots, s_{\ell/2^{i+1}}, v)$. □

Corollary 6 *With high probability, if two vertices are neighbours in G_{i+1}, then they are within distance 3 of each other in G_i.*

Corollary 6 implies that, once stage i has been finished (that is, each node of level i or higher knows its neighbours in G_i), then any node u that ends up entering level $i+1$ can discover all its neighbours in G_{i+1} (that is, compute $N_u^{(i+1)}$) by a breadth-first exploration of G_i up to distance 3, thus allowing the completion of stage $i+1$.

3.2.2 Tree partition and ball size estimates

According to Lemma 4, every node u of level $i < \log D$ is able to choose its parent of level $i+1$. It follows :

Lemma 7 *With high probability, for any i, the set of tree nodes $\{w \in T_u^i | l_u \geq i\}$ partition the set of nodes V.*

This tree partition provides an efficient way to compute estimates of ball sizes. Roughly speaking, $B_u(2^i)$ is covered by the set of subtrees (of small depth) rooted at the neighbors of u within G_i. More precisaly, we have :

Proposition 8 *With high probability, for any node u of level i, $B_u(2^i) \subset \bigcup_{v \in \tilde{N}_u^{(i)}} T_v^{(i)} \subset B_u(5.2^i)$.*

Proof. Every node $p \in B_u(2^i)$ has a parent P_p^i within a distance less than 2^{i+1}, thus $d(P_p^i, u) \leq 3.2^i$ and $P_p^i \in \tilde{N}_u(i)$. It follows that all the ancestors of level i of nodes of $B_u(2^i)$ belongs to $\tilde{N}_u^i$. The second inequality is an obvious consequence of the triangle inequality and the definitions of $\tilde{N}_u^{(i)}$ and $T_u^{(i)}$.

Now, assume that the proposed inequality holds for level i, and consider the level $i+1$ sampling. If, for each level i vertex u, there is at least one level $i+1$ (or higher) vertex v within distance 2^{i+1}, then $u \in \tilde{N}_v^{(i)}$, so that u will receive

a message from v when it enters level $i+1$ and will be able to set its $P_u^{(i+1)}$. From Lemmas 2 and 3, this situation occurs with high probability. □

From Proposition 8 and the c-bounded growth, it directly follows that computation of $\tilde{b}$ provides a c^3-approximation of balls sizes :

Lemma 9 $b_u(2^i) \leq \tilde{b}_u(2^{i+1}) \leq c^3 b_u(2^i)$.

3.3 Performance analysis

[A FAIRE]
DEGRE, MESSAGE, NB RONDES ...
If we do not pay a particular attention to the total number of messages and the number of rounds, we get

Lemma 10 *For given i, a 1-approximation of ball size $b_u(2^{i+1})$ for all $u \in S_i$ can be computed with $O(n)$ extra messages of size $O(\log n)$ and within $2i$ rounds.*

Proof. Let u be a node of level i. Take a node v within distance 2^{i+1} from u. Its ancestor s of level i is so that $d(u,s) \leq 2^{i+1}$. It follows that $d(s,u) \leq 4.2^i$ and $s \in N_u^i$.

By hypothesis, for two given nodes u and v of G, we are able to compute their distance. u asks to its neighnors of level i (at distance less than 4.2^i) the number of nodes of their subtrees that are at distance at most 2^{i+1} from u. This can be done by a BFS traversal of the subtrees. Whenever a leaf v receives such a request, it computes its distance toward u and send an acklowledgement to its parent. Internal nodes sum the total number of leaves being within $B_u(2^{i+1})$ and forwards the answer to the its parent.

A quick analysis leads to a total number of messages equal to $O(ni)$. However, there is no message whenever two copies of a same node (of consecutive levels) "exchange" a piece of information. It follows that the BFS traversal of the subtrees of the neighbors has a cost $O(n)$ in terms of messages. □

4 Small world augmentation process

While the overlay network described in the previous section provides a set of hierarchical shortcuts, they are inoperative as small world shortcuts. Indeed, greedy routing only uses the original distances of G to navigate, and therefore does not follows a link towards a higher level node if this node is not closer to the target in the original graph. However, the overlay network and the approximation of ball sizes obtained enable us to build parsimoniously, and in a distributed manner, a set of random shortcuts that make the augmented graph a small world.

We give two different constructions, both using our overlay network as a starting point. In both constructions, each node receives one additional "long-range" link. In the first construction, the links received by different nodes are

independent ; in the second construction, they are not independent, which makes it possible to significantly decrease the workload of high level nodes.

4.1 A first small-worldization process

In Algorithm 2, each node u 1) picks uniformly at random a length scale $i \in \{0, \ldots, \log D\}$ and 2) asks its ancestor s of level i to select a random shortcut for him. s then picks uniformly one of its neighbors s' with probability proportional to $|T^i_{s'}|$, and (recursively) chooses uniformly at random a leaf of $T^i_{s'}$, using the tree structure. A memory efficient distributed implementation of this step can be found in [**?**].

Algorithm 2 LONG-RANGE CONTACT(u)

Choose a level $i \in [1..\log D]$ uniformly at random
If $i > \ell_u$, send (LINKREQUEST, u, i) to P_u
If $i \leq \ell_u$, choose a node $v \in \tilde{N}^i_u$ with probability proportional to $\#T^i_v$, and send (LEAFREQUEST, i, u) to v.
Wait for reception of message (LINK, v) and set $L_u \leftarrow v$.

Upon reception of message (LINKREQUEST, w, i) :
If $i > \ell_u$, send (LINKREQUEST, w, i) to P_u
If $i \leq \ell_u$, choose a node $v \in \tilde{N}^i_u$ with probability proportional to $\#T^i_v$, and send (LEAFREQUEST, w, i) to v.

Upon reception of message (LEAFREQUEST, w, i) :
If $i = 1$, select $v \in T^0_u$ uniformly at random and send (LINK, v) to u
Otherwise, select $v \in C^i_u$ randomly, with probability proportional to $\#T^{i-1}_v$, and send (LEAFREQUEST, $w, i-1$) to v.

Theorem 11 *The resulting graph is a navigable small world. More precisely,*
- *for any two nodes u and v, the expected length of the greedy path from u to v is at most $c^7 \log_2(D) \log_2(d(u,v)) = O(\ln^2 n)$, and*
- *with high probability, for any two nodes u and v, the length of the greedy path from u to v is at most $4c^7 \ln n \log_2^2(D) = O(\ln^3 n)$.*

Proof. The long-range link that u will finally store is chosen uniformly at random from the nodes whose level i ancestor is either the level i ancestor of u, or at distance at most 3.2^i of u's level i ancestor. This set of nodes contains all nodes in $B_u(2^i)$, and is a subset of $B_u(9.2^i)$ - a set whose cardinality is at most c^4 times larger than $B_u(2^i)$. Thus, for any $v \in B_u(2^i)$, we have

$$\Pr(L_u = v) \geq \frac{1}{c^3 b_u(2^i) \log_2 D}$$

We now mirror Kleinberg's proof **??** and show that, for any destination v,

$$\Pr\left(d(L_u, v) \leq \frac{1}{2} d(u,v)\right) = \Omega\left(\frac{1}{\ln n}\right).$$

Set $d = d(u, v)$, and $i = \lceil \log_2(3d/2) \rceil$, so that $B_v(d/2) \subset B_u(2^i) \subset B_v(4d)$. Thus, the wanted probability is at least $b_v(d/2)/(c^4 b_u(2^i) \ln D)$, and we have $b_u(2^i) \leq c^3 b_v(d/2)$, so that we get the lower bound

$$\Pr\left(d(L_u, v) \leq \frac{1}{2} d(u, v)\right) \geq \frac{1}{c^6 \ln(D)} \geq \frac{1}{c^7 \ln n}.$$

Thus, the expected number of nodes the greedy algorithm visits before it finds one whose long range link cuts the distance to v in half, is at most $c^7 \ln n$, and with probability $1 - n^{-4}$, this happens before it visits $4c^7 \ln^2 n$ nodes. The two claims follow from repeating this $\log_2 D$ times.

□

4.2 An economic small-worldization process

Algorithm 3 LONG-RANGE CONTACT2(u)

Construction of long-range contacts : each node u of level $i \in [1.. \log D]$
Choose uniformly at random i^4 leaves, noted $\mathcal{L}_u^i$, in the subtrees of its neighbors N_u^i.
Spread the set $\mathcal{L}_u^i$ down T_u^i ;

Assignment of the long-range contacts : for any node u,
Choose uniformly at random a level i ;
Choose uniformly at random a long-range contact in the list $\mathcal{L}_s^i$ it received.

In the analysis of the long-distance link model, it is normally critical that the long-distance links of different nodes be chosen independently. In our first construction, they are indeed independent (conditional on the overlay network) : once the whole hierarchical tree has been built, provided some very likely properties hold, each node in the graph receives a long-distance link that is chosen independently of all others, under a probability distribution that guarantees small-world routing properties.

This independence, however, comes at a heavy price : a few nodes (those that are assigned a high level) are involved in the long-distance link choice for a very large (polynomial in N) number of other nodes (there are roughly $\Theta(\ln n)$ nodes at the highest level, and a proportion $\Omega(1/\log_2 D)$ of all link request have to be treated by one of them).

To avoid this "long-distance link selection congestion", we now propose an alternative link selection process (see Algorithm 3), in which long distance links will *not* be chosen independently - *but this will not impact the small-world routing properties.*

Once the hierarchical tree is built, each node v of each level i, instead of waiting for link requests, performs the long link selection algorithm $\ln(N)^4$ times (each time sending a (LEAFREQUEST, v, i) to one of its neighbors), and broadcasts the resulting list of nodes down its tree T_u^i. Each node $v \in T_u^i$ that, under

the previous model, would have asked u to select a long link for it (that is, sent a (LinkRequest, u, i)), simply randomly selects one of the $\ln(N)^4$ proposed nodes.

Under this new scheme, the probability distribution for the long distance link of each node is the same, but they are no longer independent - two nodes that share the same high level ancestor have a much higher probability of having the same long distance link destination. But this is not significant, because (see Theorem 11) routing paths in our original scheme are of expected lentgh $O(\ln^2 n)$, and with high probability, all are of length $O(\ln^3 n)$. Thus, each time our routing algorithm examines the long distance link of a new node, it could first check that the link is not among those it has already seen so far (same index in the same list of precomputed links), and declare it "useless" if it is not the case. The proof of Theorem 11 carries over to show that the probability of a "useless" link is only $O(1/\ln(N))$; this will, at most, result in $O(\ln(N))$ "failed" links, and the conclusions of Theorem 11 remain valid (with very slightly increased constants).

From the above discussion, we get the following :

Theorem 12 *With high probability, Algorithms 3.1 and 3 only require each node to send a polylogarithmic number of messages, and the long-distance links they compute turn G into a navigable small-world where all greedy routing paths have length $O(\log^3 n)$.*

Actually, there is no need for every "leader" node to select such a large sample of links ; since a level i node will only be required to find long distance links for vertices from its own subtree T_i, which entirely lies within distance 2^{i+1} of the root, with high probability, any routing path will only visit $O(i^2)$ nodes from this tree, so that selecting i^4 random link destinations is enough. While this does not reduce the work at the highest levels, it significantly reduces the total work of the vast majority of nodes.

5 Conclusion

One point that remains unaddressed in Kleinberg's model is an explanation of the emergence of the 2-harmonic long range links distribution in the grid. In a sense, our decentralized algorithm is light enough to be considered a first step towards a validation to real networks of Kleinberg's model. Indeed our algorithm is decentralized, applies to arbitrary bounded growth graphs and requires only polylogarithmic size memory.

However, our algorithm uses a tree-shaped hierarchy (the overlay network) which requires more coordination between the nodes than can be expected from a spontaneous network. A more flexible structure would be a better candidate to validate our model. We also have no failure detection mechanisms : in the unlikely event that the overlay network fails to remain connected, no long-distance links will cross from one component to another, and it is not clear how our current schemes could be adapted to detect such a situation.

While the "trick" of Section 4.2 very significantly reduces the maximum amount of work any node in the network will have to perform during the long range link setup (from polynomial to polylogarithmic), it does have the drawback of increasing the indegree of those nodes that are selected as potential destinations by the highest level nodes; obviously, some few nodes will have a polynomially high indegree. As a result, a large number of routing paths will go through these "unlucky" nodes, potentially resulting in network congestion.