Searching for optimal paths in long-range contact networks.

Emmanuelle Lebhar, Nicolas Schabanel

Laboratoire de l'Informatique du Parallélisme

École Normale Supérieure de Lyon
Unité Mixte de Recherche CNRS-INRIA-ENS LYON n^o 5668

Searching for optimal paths in long-range contact networks

Emmanuelle Lebhar
Nicolas Schabanel

Novembre 2003

Research Report N^o 2003-55

École Normale Supérieure de Lyon
46 Allée d'Italie, 69364 Lyon Cedex 07, France
Téléphone : +33(0)4.72.72.80.37
Télécopieur : +33(0)4.72.72.80.80
Adresse électronique : `lip@ens-lyon.fr`

Searching for optimal paths in long-range contact networks

Emmanuelle Lebhar
Nicolas Schabanel

Novembre 2003

Abstract

Since Milgram experiment in 1967, that demonstrated the ability of people to find short paths efficiently in networks, based only on their own local view of the network, different models have been proposed to study the "small world phenomenon". In 2000, Kleinberg shows that most of the previously known models for small world fail to capture an important feature of the phenomenon: no local information based (i.e., decentralized) algorithm can find short paths in these graphs, when they exists. He then introduces a model composed of a lattice (representing the local acquaintances) augmented with directed links (symbolizing long-range contacts) distributed harmonically. He proposes to model people behavior by the greedy algorithm that forwards the message to the contact (local or long-range) of the current holder, which is the closest to its destination. He shows that this greedy algorithm computes paths of expected length $\Theta(\log^2 n)$ between any pair of nodes. The present paper questions the choice of the greedy strategy to model social behavior. We proposes a new strategy in which nodes consult their acquaintances near by, before deciding where to forward the message. Our algorithm presents the same computational characteristics as Kleinberg's original algorithm: uses $\Theta(\log n)$ bits of memory and visits $O(\log^2 n)$ nodes. However, it computes much shorter paths, of expected length $O(\log n(\log \log n)^2)$, between any pair of nodes. This algorithm demonstrates that consulting their acquaintances near by, leads to considerable improvements in performances. It also shows that this consultation is less and less useful as one gets closer to the target. As far as we know, this is the first algorithm to "break the $\Theta(\log^2 n)$ barrier" for the paths length in Kleinberg's network.

Keywords: algorithms on random structures, routing, small world and social behavior models.

Résumé

Depuis l'expérience de Milgram de 1967, qui démontre que les individus trouvent efficacement des courts chemins dans de grands graphes en utilisant uniquement leur vision locale du graphe, plusieurs modèles ont été proposés pour étudier ce phénomène de "petit monde". En 2000, Kleinberg a montré que la plupart des modèles précédemments connus manquaient une caractéristique importante du phénomène: aucun algorithme décentralisé ne pouvait trouver les courts chemins lorsqu'ils existaient. Il a alors introduit un modèle composé d'une grille carrée (représentant les connaissances locales) augmentée de liens dirigés (symbolisant les contacts distants) distribués harmoniquement. Il propose de modéliser le comportement des individus par l'algorithme glouton qui passe le message au contact (local ou distant) du possesseur actuel du message qui est le plus proche du destinataire. Il montre que cet algorithme glouton calcule un chemin de longueur moyenne $\Theta(\log^2 n)$ entre toute paire de noeuds. Nous nous poserons la question de la pertinence du choix de la stratégie gloutonne pour modéliser les comportements sociaux. Nous proposons une nouvelle stratégie dans laquelle les noeuds consultent leurs connaissances avant de décider à qui envoyer le message. Notre algorithme présente les mêmes caractéristiques de complexité que l'algorithme original de Kleinberg: il utilise $\Theta(\log n)$ bits de mémoire et visite $O(\log^2 n)$ noeuds. Il calcule des chemins plus courts, de longueur moyenne $O(\log n(\log \log n)^2)$, entre toute paire de noeuds. Cet algorithme montre que consulter son voisinage conduit à des améliorations de performances, il montre aussi que cette consultation est de moins en moins utile lorsque l'on s'approche de la destination.

Mots-clés: algorithmes sur les structures aléatoires, routage, petit monde et modèles de comportements sociaux.

1 Introduction

1.1 Motivations and previous work

The small world experiment. This experiment was introduced by Milgram in 1967 [7]. People were asked to forward a letter to one of their acquaintance they believe to be the closest to the target person, described by its name, location, and occupation. The striking result was that, when the letter finally reached its target, only 5 to 6 steps were needed. This is a typical small world phenomenon. Since this experiment, a lot of social graphs (such as the co-author graph, the web graph,...) were discovered to share similar properties : a very small diameter (typically poly-logarithmic in the size of the network) and the existence of short paths between random nodes, that can be found very efficiently, based only on the local view of the network.

Models for the small world phenomenon. Newman in [8, 9] presents an interesting state of the art on the matter. Several models have been proposed to capture what are the specific properties that makes a graph a small world. Watts and Strogatz [11] observed that most of the small world graphs are locally strongly interconnected : two nodes with a common neighbor are very likely to be connected to each other. They proposed a model based on a ring where each nodes is connected to its k closest neighbors on the ring, and where each edge is uniformly rewired with higher and higher probability. This model shows that, as randomness increases, the diameter gets smaller and smaller, until the network gets disconnected. More tractable variants of this model, that replace the rewiring process by edge percolation [10] or by the addition of a random matching [3], have been studied theoretically, in particular, their diameter. But these models failed to capture the specific nature of small world, as demonstrates Kleinberg in his seminal paper [5]. He shows that, for these models, there does not exist any algorithm that can find short path (with poly-logarithmic length in the size of the network) based only on local information, even if the diameter is poly-logarithmic. He then introduces a new model : a 2-dimensional lattice augmented with random directed links. The 2-dimensional lattice represents the underlying geographic relationships between the people. In addition to its four *local* neighbors, each node $\mathbf{u}$ is the origin of a directed edge pointing to its *long range contact* $\mathbf{v}$, chosen randomly according to the s-harmonic distribution, i.e., with probability proportional to $1/\delta(\mathbf{u},\mathbf{v})^s$, where $\delta(\mathbf{u},\mathbf{v})$ is the lattice (Manhattan) distance between $\mathbf{u}$ and $\mathbf{v}$. Kleinberg introduces the notion of decentralized algorithm to model human social behavior, such as passing a letter from acquaintance to acquaintance to reach as fast as possible its destination [7], or to model the search for informations in the web by clicking on hyperlinks or on back-button [6]. A *decentralized algorithm* constructs a path between any random pair of nodes $\mathbf{u}$ and $\mathbf{v}$, based only on local information, that is to say, with the knowledge of the underlying lattice structure, but constrained to explore only neighbors of already visited nodes, and ignoring all of the long-range contacts of the nodes it did not go through. Kleinberg demonstrates that when $s \neq 2$, no decentralized algorithm can find a poly-logarithmic length path in the network. He also describes the very simple greedy algorithm that follows, from each node, the link to its closest contact to the target, among the five possible, until it reaches the target. He shows that if $s = 2$, this algorithm computes a path of expected length $\Theta(\log^2 n)$, between any random pair of nodes [5, 1]. This result demonstrates that there is more to the small effect than simply the existence of short paths, and that the algorithmic nature of the experiment has to be considered. Kleinberg's model and result have been generalized to an arbitrary d-dimensional lattice augmented with long range contacts s-harmonically distributed in [1] : only $s = d$ defines a small world, in the decentralized algorithm sense. And, when $s = d$, Kleinberg's greedy algorithm computes path of expected length $\Theta(\log^2 n)$ between any pair of nodes, somehow surprisingly independent of the dimension. The authors of [1] conclude that the d-harmonic distribution of the long-range links create a strong correlation with the dimension of the lattice, and then that one could equivalently focus on the one-dimensional case, without significant changes in the properties of the network. Undirected versions of this graph, based on edge percolation, have been studied in [2, 4].

On the algorithmic perspective. The works in [5, 1] raise several important questions : how should we define a small world algorithmically ? How should we limit decentralized algorithms to match social behavior, such as people consulting their acquaintances before sending the letter ? Is Kleinberg's greedy algorithm the most natural decentralized algorithm ? Does it compute optimal paths ? Is it an approximation ? Can we give a tight bound on the diameter of the Kleinberg's network ? More generally, is this algorithm *"canonical"* to define small world graphs ? These last points raise another important algorithmic perspective : since most of the "real-life" graphs share small world properties, practical interest dictates to design algorithms that take advantage of these specific structures.

1.2 Our contribution

We focus on the one-dimensional Kleinberg's small world network model. We study a new decentralized algorithm that computes a path of expected length $O(\log n(\log\log n)^2)$ between any pair of nodes in the network, and visits $O(\log^2 n)$ nodes to build it. As far as we know, this is the first algorithm to "break the $\Theta(\log^2 n)$ barrier" in Kleinberg's network. This demonstrates that Kleinberg's greedy algorithm computes paths arbitrarily longer than the optimal. Interestingly enough, the average length of the paths between random nodes in Kleinberg's original network with *directed* long-range contacts and bounded out-degree, is very close to $\Theta(\log n/\log\log n)$, the expected length of the shortest path in the *undirected* version, with unbounded degree, of this network studied in [4]. This may allow to reduce the size of routing table in application using Kleinberg's structure (as in [12]).

2 Model and main results

The network. We consider the one-dimensional variant proposed in [1] of the long-range contact network model introduced by Kleinberg in [5]. The network is an augmented ring of $2n+1$ nodes, numbered from $-n$ to n. In addition to its two neighbors in the ring (its *local contacts*), each node $\mathbf{u}$ has an extra directed link pointing towards a node $\mathbf{v}$ ($\mathbf{u}$'s long-range contact), chosen independently according to the 1-harmonic distribution, i.e., with probability proportional to $1/\delta(u,v)$, where $\delta(u,v)$ is the distance between u and v on the ring. The exact probability that $\mathbf{v}$ is $\mathbf{u}$'s long range contact is $1/(2H_n\delta(u,v))$, where $H_n = \sum_{i=1}^{n} 1/i$ is the normalizing constant. From now on, the distance between two nodes refers to their distance on the underlying ring, and log stands for the logarithm base 2, while ln denotes the natural logarithm, base e. Note that $\ln(n+1) < H_n < \ln n + 1$.

Kleinberg also defines variants of its model, where nodes are connected by an undirected edge to their lattice neighbors at distance $\leq k$, or are given k' long-range contacts i.i.d. for given constants k and k'. These variants do not change significantly the properties of the network. Our analysis and results are also identical on these variants up to a constant factor. Therefore, the present paper focuses on the $k = k' = 1$ case described above.

Decentralized algorithms with bounded memory. A *path* is a sequence of links (local or long-range) between neighboring nodes from a source to a target ; its length is the number of links used ; a path is said *local* if it is composed of local links only. We study algorithms that compute a path to transmit a message from its source to its target, along the edges and arcs of the network. Following Kleinberg's definition, such an algorithm is *decentralized* if it navigates through the network using only local informations to compute the path. In particular, it has the knowledge 1) of the underlying lattice structure (here, the ring), 2) of the coordinates of the target, and 3) of the nodes it has previously visited as well as their long-range contacts. But, crucially, 4) it cannot visit a node which is not the local or long-range contact of a previously visited node, and 5) does not know the long-range contact of any node that has not yet been visited. However, 6) the algorithm (but not the path it computes) is authorized to travel across any directed links it has already followed. As pointed out in [6], this is a crucial component of human ability to find short paths : one can interpret point 6) as a web user pushing the back button, or an individual returning the letter to its previous holder.

If this definition is sufficient to prove that, among the small world model candidates with poly-logarithmic diameter in the size of the network, some do not have any decentralized algorithm that computes poly-logarithmic length path, (such as [11, 3] or the Kleinberg network with an s-harmonic distribution for the long-range contacts, when s differs from the dimension of the underlying lattice [5, 1]), one should however limit the memory of the algorithm. It is unreasonable to assume that social individuals may remember all the previous holders of the message. We thus refine Kleinberg's definition as follows : an algorithm is a *decentralized algorithm with bounded memory* if it is decentralized and keeps only a poly-logarithmic number (in the size of the network) of bits in memory.

As far as we know, only one decentralized algorithm has been studied so far, essentially in [5, 1] : the "simplest" greedy algorithm that always transmits the message to the closest neighbor (local or long-range) of the current holder to the target, in the sense of the distance. In [5] and [1], this algorithm is shown to compute a path of expected length $\Theta(\log^2 n)$, independly of the dimension of the network. Note that this algorithm uses exactly $\Theta(\log n)$ bits of memory (to store the target coordinates). However it is unlikely that, in Milgram experiment, individuals just forward the message to the first acquaintance that comes to their mind. It is more likely that they contact their immediate friends to discuss with them who would be the closest to the target, among their friends. The same is true for the web : when searching for some data, it is usually much more efficient to try several links before deciding which one to follow. This raises several questions : is Kleinberg's greedy algorithm the most natural algorithm ? should we use it to test the small world property ? In other word, is it "canonical" ? Also, from an algorithmic point of view, does it compute optimal paths ? or an approximation of the optimal paths ? What is the diameter of Kleinberg's small world network ?

Main result. As suggested in [1], we focus on the one-dimensional network. The following theorem shows that with $\Theta(\log n)$ bits of memory, one can find a path of expected length $O(\log n(\log\log n)^2)$ from any source to any target, while visiting $O(\log^2 n)$ nodes.

Theorem 1 *There is a decentralized algorithm $\mathcal{A}$ with $\Theta(\log n)$ bits of memory so that, for any pair of nodes i and j, $\mathcal{A}$ computes a path from i to j of expected length $O(\log n(\log\log n)^2)$, and visits $O(\log^2 n)$ vertices to compute this path, on expectation.*

The principle of our algorithm is the following : the closer one gets to the target, the less useful are the long-range contacts. As long as the message is "far" from the target, the algorithm looks around for the best long-range contact among the nodes a few links away (local or long-range), and gets to it ; once the message is "close" to the target, checking around the neighborhood gets useless, and we use Kleinberg greedy algorithm to finally reach the target.

As far as we know, this is the first algorithm to break the $\Theta(\log^2 n)$ barrier in Kleinberg's graph. This theorem answers some of the questions asked above. First, Kleinberg's greedy algorithm computes paths arbitrarily longer than the optimal. Second, with the same amount of memory, that is to say, with the same hypothesis on the computational power of a social individual, one can compute shorter paths, following a very natural principle.

The following section presents the algorithm and its analysis in details. The proofs of the pure probabilistic lemmas is postponed to section 4.

3 The algorithm

Description of the algorithm. Without loss of generality, we assume that the target is $\mathbf{0}$. We denote by $\mathbf{x}$ the current position of the message, and by x its distance to the target. If $x < \log^2 n$, we apply Kleinberg's greedy algorithm that reaches the target in $O(\log n \log x) = O(\log n \log\log n)$ steps on expectation (see [5]).

From now on, we assume that $x \geq \log^2 n$. Let us introduce the following definitions.

Definition 1 *A link (local or long-range) from a node* $\mathbf{u}$ *to a node* $\mathbf{v}$ *is called a* jump *if* $\mathbf{v}$ *is strictly closer to the target than* $\mathbf{u}$*. Every node* u *(but the target) is the origin of at least one (local) jump and, with some probability, of a second (long-range) one.*
A node $\mathbf{v}$ *is said to* be h jumps away *from* $\mathbf{u}$*, if there is a path only made of jumps from* $\mathbf{u}$ *to* $\mathbf{v}$*.*

Each step of the algorithm proceeds in two phases : 1) the exploration phase that basically looks for the closest node to the target, among $\Theta(\log n)$ nodes less than $\Theta((\log n \log\log n)/\log x)$ jumps away from $\mathbf{x}$; and 2) the routing phase, that forwards the message to this node. Proposition 2 shows that each step divides the distance of the message to its target by 2 with constant positive probability (independent of n and $\mathbf{x}$).

An important fact is that each exploration phase is independent of the previous ones. Since the algorithm only follows jumps towards the target, and forwards the message to the visited node which is the closest to the target, during every new exploration phase, the algorithm explores only unvisited nodes. We can then conveniently assume that their long-range contact are independently generated at this moment.

The key to Proposition 2 is to guarantee that $\Omega(\log n)$ distinct nodes are visited during the exploration phase, with constant positive probability. We have to deal with three facts : 1) we have less and less long-range jumps as we get closer to the target, since it is more and more likely to "jump" over the target ; 2) the risk of visiting twice a node increases with the number of long-range jumps we take ; 3) we seek for a "good" node, close to the target, but also as close as possible from $\mathbf{x}$ (in order to minimize the path length). Fact 1 implies that we will have to look further and further as we get closer to the target. In order to find a "good" node, as close as possible from $\mathbf{x}$ (Fact 3), the best strategy is to choose the smallest h so that one can reach $\log n$ nodes in h jumps. Unfortunately, the likelihood of visiting twice a node increases too fast (Fact 2) and this strategy does not work. We obtain then the trade-off below.

Let $h = \Theta((\log n \log\log x)/\log x)$ and $k = \Theta((\log n \log\log n)/\log x)$ (their precise values will be given later). During the exploration phase, the algorithm first explores all the vertices less than h jumps away from $\mathbf{x}$; let A be the set of these vertices. Then, the algorithm visits all the nodes at distance less than k towards the target, from the nodes in the "boundary" of A, that is to say from the nodes at h jumps exactly from $\mathbf{x}$; let B be the set of these nodes. If, at any time, the algorithm visits a node at distance $\leq x/2$ from the target, it stops the exploration process and forwards the message to it. Otherwise, it forwards the message to the closest node to the target, among $A \cup B$ and their long-range contacts.

The values of h and k claimed above lead to the expected result :

Proposition 2 *There exists a constant $c_1 > 0$, independent of n and $\mathbf{x}$, such that with probability c_1, there is a node in B whose long-range contact is at distance $\leq x/2$ from the target.*

Analysis of the algorithm. The proof of this proposition relies on the following lemmas.

Lemma 3 ([5]) *Given $s > 0$, there is a constant $c_2 > 0$, such that, for any subset S of $s \cdot \log n$ vertices at distance in $(x/2, x]$ to the target, one vertex in S (at least) has its long-range contact at distance $\leq x/2$ to the target, with probability c_2.*

Proof. This lemma is essentially proved in [5]. Let $\mathbf{u}$ be a vertex at distance $u \in (x/2, x]$ to the target. The probability that $\mathbf{u}$'s long-range contact is at distance $\leq x/2$ to the target, is : $1/(2H_n)\sum_{i=u-x/2}^{u+x/2} 1/i \geq 1/(2H_n)\int_{u-x/2}^{u+x/2} dt/t \geq \ln 2/(2H_n)$. The probability that all the long-range contacts of the nodes in S are at distance $> x/2$ to the target, is then less than $\left(1 - \ln 2/(2H_n)\right)^{s \log n} \leq e^{-s/2} < 1$. □

The following lemma is the key to ensure that when following the long-range jumps, we do not revisit previously visited node during the exploration step.

Lemma 4 *Let $\mathbf{u}$ a node at distance $u \in (x/2, x]$ to the target, $\mathbf{v}$ its long-range contact, S a set of f forbidden nodes, and q an integer. Assuming that $\mathbf{v}$ is at distance $< u$ from the target, the probability that $\mathbf{v}$ is at distance $\geq q$ from any node of S is $\geq 1 - H_{2fq-1}/H_{x-1}$.*

Proof. Let $\mathcal{E}$ be the event that $\mathbf{v}$ is at distance $\geq q$ from any node of S, given that $\mathbf{v}$ is at distance $< u$ from the target. $\mathcal{E}$ can be seen as the event that $\mathbf{v}$ does not belong to the f intervals of length $2q$ centered on the vertices of S. We use an important property of the long-range link length distribution to bound the probability of $\mathcal{E}$: the probability that $\mathbf{v}$ is at distance δ from $\mathbf{u}$ is decreasing with δ. Therefore, the probability of $\mathcal{E}$ is minimized when the f intervals are all disjoint, as close as possible to $\mathbf{u}$, and between $\mathbf{u}$ and the target. The probability of $\mathcal{E}$ is then greater than the probability that v is at distance $\geq 2fq$ from $\mathbf{u}$, given that $\mathbf{v}$ is at distance $< u$ from the target. We conclude that : $\Pr \mathcal{E} \geq \frac{1}{H_{2u-1}} \sum_{i=2fq}^{2u-1} \frac{1}{i} \geq 1 - \frac{H_{2fq-1}}{H_{x-1}}$. □

We now study the structure of the set A in details. As pointed out above, every node $\mathbf{u}$ in the network is the origin of one (local) jump or two (one local and one long-range) jumps. The probability that a node $\mathbf{u}$, at distance u, is the origin of two jumps is $\alpha_u = H_{2u-1}/(2H_n)$ (the probability that its long-range contact is closer to the target than it is). A can thus be seen as a partial binary tree of height h, where each node $\mathbf{u}$ has one child with probability $1 - \alpha_u$, and two children with probability $1 - \alpha_u$ (note that some nodes may appear twice in the tree, but Lemma 4 will guarantee that this will not happen with constant probability). Let us color local jumps in blue and long-range jumps in red in the tree. Let R be the number of red jumps in the tree. The tree is now the union of exactly $(R+1)$ blue branches. Thus A can be decomposed in $R+1$ intervals (i.e., local paths) of length $\leq h$ on the ring, connected to each other by R long-range jumps. *The set $A \cup B$ is thus exactly the union of $R+1$ intervals of length $\leq h+k$, connected to each other by long-range jumps.*

We now count these intervals, that is to say, the number B_h of branches in the tree at level h $(B_h = R+1)$.

Lemma 5 *Take $h = (\log H_x - \log\log H_n)/\log(1 + H_{x-1}/(2H_n))$, ($h = O(\log n \log\log x/\log x)$). Assuming that the exploration phase is completed, the expected value of B_h is : $\mathbb{E}[B_h] = \Theta(\log x/\log\log n)$; and there exist three constants $\mu > \lambda > 0$ and $c_3 > 0$, independent of n and x, such that with probability c_3 : $\lambda \log x/\log\log n \leq B_h \leq \mu \log x/\log\log n$.*

Proof. Since the exploration phase is completed, all nodes in A are at distance between $x/2$ and x from the target, thus for all $\mathbf{u}$ in A, $H_{x-1}/(2H_n) \leq \alpha_u \leq H_{2x-1}/(2H_n)$. Let $\alpha^- = H_{x-1}/(2H_n)$ and $\alpha^+ = H_{2x-1}/(2H_n)$. Let B_h^- (resp., B_h^+) the number of branches at level h in the branching process that starts with one node, and where each node has one child with probability $1-\alpha^-$ (resp., $1-\alpha^+$), and two children with probability α^- (resp., α^+). B_h^+ stochastically dominates B_h which in turn stochastically dominates B_h^-.

To improve the readability of the algorithm, we have transfered the analysis of the branching processes to the separate section 4. Lemma 7 from section 4, states that : $\mathbb{E}[B_h^-] = (1+\alpha^-)^h$ and $\mathbb{E}[B_h^+] = (1+\alpha^+)^h$. Therefore, $(1+\alpha^-)^h \leq \mathbb{E}[B_h] \leq (1+\alpha^+)^h$.

From Markov bound, for any $\nu > 1$, $\Pr\{B_h^+ \leq \nu(1+\alpha^+)^h\} \geq 1 - 1/\nu$. Lemma 8 states that : there are two constants $\lambda > 0$ and $c > 0$, independent of α^- and h, i.e., of $\mathbf{x}$ and n, such that : $\Pr\{B_h^- \geq \lambda(1+\alpha^-)^h\} \geq c$. Therefore :

$$\begin{aligned}\Pr\left\{\lambda(1+\alpha^-)^h \leq B_h \leq \frac{2}{c}(1+\alpha^+)^h\right\} &\geq \Pr\left\{B_h^- \geq \lambda(1+\alpha^-)^h \text{ and } B_h^+ \leq \frac{2}{c}(1+\alpha^+)^h\right\}\\ &\geq c/2 > 0.\end{aligned}$$

But, with the claimed value of h, $(1+\alpha^-)^h = H_x/\log H_n$, and there exists a constant $a > 0$, independent of n and $\mathbf{x}$, such that $(1+\alpha^+)^h \leq aH_x/\log H_n$. This concludes the proof. □

We can now accomplish the last step to prove Proposition 2.

Corollary 6 *Take $k = \log n \log\log n/\log x$. There exists three constants $\lambda > 0$, $c_4 > 0$, and n_0, such that for $n \geq n_0$, with probability c_4, there are more than $\lambda \log n$ distinct nodes in B.*

Proof. As pointed out earlier, $A \cup B$ is the union of B_h intervals of length $\leq h+k$, connected to each other by long-range jumps. The probability p that one of these intervals does not intersect any of the other, is bounded by Lemma 4 : $p \geq 1 - \frac{H_{2B_h \cdot (h+k)}}{H_x - 1}$.

Thus all nodes in $A \cup B$ are visited only once with probability $\geq (1 - \frac{H_{2B_h \cdot (h+k)}}{H_{x-1}})^{B_h}$. Since the computation of B_h is the result of a branching process *independent* of the effective positions of the long-range jumps, we are free to use Lemma 5 to bound the collision probability. There exist three constants $0 < \lambda < \mu$ and $c_3 > 0$, independent of n and $\mathbf{x}$, such that : $\lambda \log x/\log\log n \leq B_h \leq \mu \log x/\log\log n$, with probability c_3. Note that $h+k \leq 2k \leq 2\log n \log\log n/\log x$. There exists $n_0 > 0$ such that for $n \geq n_0$, $H_{x-1} \geq H_{\log^2 n} \geq \frac{3}{2} H_{4\mu \log n}$. Then, for $n \geq n_0$, the probability that no node in $A \cup B$ is visited twice is :

$$\geq \left(1 - \frac{H_{4\mu \log n}}{H_{x-1}}\right)^{\mu \frac{\log x}{\log\log n}} \geq c_3(1-2/3)^{3/2} > 0$$

Thus with probability $\frac{c_3}{3^{3/2}}$, the number of distinct nodes in B is $k \cdot B_h \geq \lambda \log n$. □

Proof of Proposition 2. Let $\lambda > 0$, and $c_4 > 0$ from Corollary 6. With probability c_4, there are $\lambda \log n$ distinct nodes in B, and none of their long-range contacts have been visited yet. Let $c_2 > 0$ from Lemma 3. Since both events are independent, with constant probability $c_2 c_4 > 0$, at least one node in B has its long-range contact at distance $\leq x/2$ to the target. □

We now conclude with the proof of the theorem for $d = 1$.

Proof of Theorem 1. Assume that we want to transmit a message from $\mathbf{u}$ to $\mathbf{0}$. Let $\mathbf{x}$ denote the current message holder of the message. First recall that at the end of each exploration step, the algorithm selects the closest node in $A \cup B$ from the target, and that the set $A \cup B$ grows by jumps towards the target ; therefore, every exploration step visits unexplored nodes, and each exploration step is independent of the previous ones.

We divide the execution of $\mathcal{A}$ in $\log n$ phases ; the execution is in phase K, $0 < K < \log n$, if $2^K < \delta(\mathbf{x}, \mathbf{0}) \leq 2^{K-1}$. We say that an exploration step in phase K succeeds if it leads to a phase $\leq K - 1$. Let Y_K and Z_K, the random variables for the number of visited nodes in phase K, and for the length of the path along which the message is routed in phase K, respectively.

Suppose that we are in phase K, with $\log n < K \leq 2 \log \log n$. According to Proposition 2, each exploration step succeeds with probability $\geq c_1$. Each exploration step visits $(h + k)\mathbb{E}[B_h]$ nodes in expectation, and routes the message, along a path of length $\leq h + k$ towards the target. Then, $\mathbb{E}[Y_K] \leq (h + k)\mathbb{E}[B_h]/c_1 = O(\log n)$ and $\mathbb{E}[Z_K] \leq (h + k)/c_1 = O(\log n \log \log n/K)$.

Once $K \leq 2 \log \log n$, the algorithm uses Kleinberg's greedy algorithm. From [5], we have : $\mathbb{E}[Y_K] = \mathbb{E}[Z_K] = O(\log n)$.

The expected length of the path computed by our algorithm from $\mathbf{u}$ to $\mathbf{0}$ is :

$$\begin{aligned} \sum_{K=0}^{\log n} \mathbb{E}[Z_K] &\leq \sum_{K \leq 2 \log \log n} O(\log n) + \sum_{K > 2 \log \log n} O(\log n \log \log n/K) \\ &= O\big(\log n (\log \log n)^2\big). \end{aligned}$$

The expected number of nodes visited by our algorithm is :

$$\sum_{K=0}^{\log n} \mathbb{E}[Y_K] = O(\log^2 n).$$

This algorithm just needs $\Theta(\log n)$ bits to keep : the location of the target, the state of the stack during the depth-first search of $A \cup B$, and both location and state of the stack for the current best node in $A \cup B$. □

One could also use the same technics as in [4], to show that the total number of exploration steps in our algorithm is bounded by $O(\log n)$ with probability $1 - o(1/n)$. Unfortunately, this does not yield a bound with high probability on the length of the path computed, nor on the diameter of the graph, since the path length within an exploration depends on the phase number, which is the key of our result. Finding a better bound than $O(\log^2 n)$ with high probability, on the diameter of Kleinberg's network, remains an open question.

4 Analysis of the branching process

Let $0 < \alpha \leq 1$. Consider the following branching process : start with one node ; during the h-th step, every node at level $h - 1$, is given one child with probability $1 - \alpha$, and two children with probability α. The following lemma evaluates the expected number of branches at level h and its variance, in the resulting binary tree. This probabilistic lemma is the key to Lemma 5.

Let B_h be the random variable for the number of branches at level h.

Lemma 7 $\mathbb{E}[B_h] = (1 + \alpha)^h$ *and* $\operatorname{Var}[B_h] \leq \frac{1-\alpha}{1+\alpha}(1 + \alpha)^{2h}$.

Proof. We proceed by induction. For $h = 0$, $\mathbb{E}[B_0] = 1$ and $\mathrm{Var}[B_0] = 0 \leq \frac{1-\alpha}{1+\alpha}$. Suppose now $h > 0$. The expected number of children of every node at level $h-1$, is $(1+\alpha)$. The expected number of nodes at level h, is thus $(1+\alpha)\mathbb{E}[B_{h-1}]$. Therefore, $\mathbb{E}[B_h] = (1+\alpha)^h$.

We now consider the variance. With probability $1-\alpha$, the root of tree has one child ; let X be the random variable for the number of branches of this only child. With probability α, the root has two children ; let Y and Z be the random variable for the number of branches for each of them. X, Y and Z are independent variables distributed identically to B_{h-1}. In particular, the expectation and variance of X, Y and Z are equal to $\mathbb{E}[B_{h-1}]$ and $\mathrm{Var}[B_{h-1}]$, respectively. Now,

$$\mathbb{E}[B_h^2] = (1-\alpha)\mathbb{E}[X^2] + \alpha\mathbb{E}[(Y+Z)^2] = (1+\alpha)\mathbb{E}[B_{h-1}^2] + 2\alpha\mathbb{E}[B_{h-1}]^2,$$

since Y and Z are independent. And as $\mathbb{E}[B_h] = (1+\alpha)\mathbb{E}[B_{h-1}]$, we have :

$$\begin{aligned}\mathrm{Var}[B_h] = \mathbb{E}[B_h^2] - \mathbb{E}[B_h]^2 &= (1+\alpha)(\mathbb{E}[B_{h-1}^2] - \mathbb{E}[B_{h-1}]^2) + \alpha(1-\alpha)\mathbb{E}[B_{h-1}]^2 \\ &= (1+\alpha)\,\mathrm{Var}[B_{h-1}] + \alpha(1-\alpha)(1+\alpha)^{2h-2}.\end{aligned}$$

By induction, $\mathrm{Var}[B_{h-1}] \leq \frac{1-\alpha}{1+\alpha}(1+\alpha)^{2h-2}$, so :

$$\mathrm{Var}[B_h] \leq \tfrac{1-\alpha}{1+\alpha}(1+\alpha)^{2h-2}(1+2\alpha) \leq \tfrac{1-\alpha}{1+\alpha}(1+\alpha)^{2h}.$$

□

We now show that with constant positive probability, the number of branches in the branching process is close to its expectation up to a constant factor.

Lemma 8 *There exist three constants $\mu > \lambda > 0$ and $c_3 > 0$, independent of α and h, such that with probability c_3, $\lambda\mathbb{E}[B_h] \leq B_h \leq \mu\mathbb{E}[B_h]$.*

Proof. For the sake of readability, let B and E refer to B_h and $\mathbb{E}[B_h]$. For $\mu > \lambda > 0$, let $q = \Pr\{\lambda\, E \leq B \leq \mu E\}$. We will show that q is always greater than a positive constant for suitable values of λ and μ. We rewrite E as follows :

$$E = \sum_{0\leq k\leq\lambda E} kPr(B=k) + \sum_{\lambda E\leq k\leq\mu E} kPr(B=k) + \sum_{\mu E\leq k} kPr(B=k).$$

Let us denote these three sums respectively S_1, S_2 and S_3. Clearly, $S_1 \leq \lambda E$ and $S_2 \leq \mu E q$.

We bound the third sum with Cauchy-Schwarz inequality :

$$S_3 \leq \sqrt{\sum_{\mu\mathbb{E}\leq k} k^2 \Pr\{B=k\}}\sqrt{\sum_{\mu\mathbb{E}\leq k} \Pr\{B=k\}}.$$

According to Markov bound, $\sum_{\mu E\leq k} Pr(B=k) \leq 1/\mu$. As $\sum_{\mu E\leq k} k^2 \Pr\{B=k\} \leq \mathbb{E}[B^2]$, we get $S_3 \leq \sqrt{\mathbb{E}[B^2]/\mu}$.

Finally, $E \leq \lambda E + \mu q E + \sqrt{\mathbb{E}[B^2]/\mu}$, that is to say :

$$q \geq \frac{1-\lambda}{\mu} - \frac{\sqrt{\mathbb{E}[B^2]}}{\mu^{3/2}E}.$$

According to Lemma 7, $\mathrm{Var}[B] \leq \frac{1-\alpha}{1+\alpha}(1+\alpha)^{2h} = \frac{1-\alpha}{1+\alpha}E^2 \leq E^2$, since $0 < \alpha \leq 1$. Thus :

$$q \geq \frac{1-\lambda}{\mu} - \frac{2}{\mu^{3/2}}$$

Setting $\lambda = 1/2$ and $\mu = 18$, gives $q \geq 1/108 =_{\mathrm{def}} c_3 > 0$, which concludes the proof. □

5 Conclusion

We prove that the expected path length between any pair of nodes in Kleinberg's network is $O(\log n (\log \log n)^2)$. We don't know at this point how to get the same bound with high probability. In particular, the question of the diameter of Kleinberg's network remains an open question. We currently study the generalization of this algorithm to the d-dimensional network which should extend the same bound on the expected path lengths. This raises the question of the influence of the dimension of the underlying lattice in Kleinberg's network.

Our algorithm demonstrates that, consulting its acquaintances before forwarding the message (or snooping around web pages before choosing which hyperlinks to follow) allows to compute very short paths, but is also less and less useful as one gets closer to its target. This reinforces our feeling that this network model is very convincing to model the small world phenomenon. An interesting experiment would be to get, from the real social data, how many long-range links are used by social agents as the distance to the target decreases : this might be an interesting criterion to decide which algorithm is closer to human behavior.

A crucial question remains open however : what "happens" in the graph structure with the d-harmonic distribution, which makes decentralized algorithms work ?

Références

[1] L. Barrière, P. Fraigniaud, E. Kranakis, and D. Krizanc. Efficient routing in networks with long range contacts. *15th International Symposium on Distributed Computing (DISC '01)*, LNCS 2180 :270–284, 2001.

[2] I. Benjamini and N. Berger. The diameter of long-range percolation clusters on finite cycles. *Random Structures and Algorithms*, 19(2) :102–111, 2001.

[3] B. Bollobás and F.R.K. Chung. The diameter of a cycle plus random matching. *SIAM J. Discrete Math.*, 1 :328–333, 1988.

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

[5] J. Kleinberg. The small-world phenomenon : an algorithmic perspective. *Proc. 32nd ACM Symposium on Theory of Computing*, pages 163–170, 2000.

[6] J. Kleinberg. Small-world phenomena and the dynamics of information. *in T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.) Advances in Neural Information Processing Systems 14. MIT Press, Cambridge, MA.*, 14, 2002.

[7] S. Milgram. The small world problem. *Psychology Today*, 61(1), 1967.

[8] M. E. J. Newman. Models of the small world. *J. Stat. Phys.*, 101, 2000.

[9] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2) :167–256, 2003.

[10] M. E. J. Newman and D. J. Watts. Renormalization group analysis of the small-world network model. *Phys. Lett. A*, 263 :341–346, 1999.

[11] D. Watts and S. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(440–442), 1998.

[12] H. Zhang, A. Goel, and R. Govindan. Using the small-world model to improve freenet performance. *Proceedings of IEEE INFOCOM*, 2002.