Social Networks

Small world phenomenon in a social network:

- short chains exist in the network of acquantances.
- people are able to find these short chains knowing so little about the target individual.

Two questions:

- Why should there exist short chains of acquaintances linking together arbitrary pairs of strangers?
- Why should arbitrary pairs of strangers be able to find short chains of acquaintances that link them together?

Models which will explain both these phenomena.

A Network Model

A directed graph:

- Nodes (individuals): a set of lattice points in a $n \times n$ square – $\{(i,j) : i \in \{1,2,\ldots,n\}, j \in \{1,2,\ldots,n\}\}$.

- Lattice distance between two nodes (i,j) and (k,l): $d((i,j),(k,l)) = |k-i| + |l-j|$.

- local edges: every node u has a directed edge to every other node within lattice distance p, where p is a fixed constant.

- long-range edges: every node u has a directed edge to q other nodes chosen by independent trials: the ith directed edge from u has endpoint v with probability proportional to $1/(d(u,v))^r$; $q \geq 0$, $r \geq 0$ are fixed constants. $r = 0$ is called a uniform distribution, $r = 1$ is an inverse distribution, and $r = 2$ is a quadratic inverse distribution

There is a similar model proposed by Strogatz and Watts in a 1998 article in Nature. This model consists of a ring of nodes where every node connects to all nodes within a fixed distance and also to a few random nodes. This model of the network demonstrated that with just a few random connections messages could travel fast, even in such a large network. They suggested that these networks exist not only among human beings but in other types of networks such as power grids and the human brain. However, Kleinberg proved that this mathematical model didn't allow for an efficient computer algorithm to find these paths. (http://www.news.cornell.edu/Chronicle/00/8.31.00/six_degrees_sep.html)

Decentralized algorithms

Given two arbitrary nodes s and t the goal is to transmit a message from s to t in as few steps as possible.

A decentralized algorithm - message is passed sequentially from a current message holder to one of its local or long-range contacts, using only local information.

We assume that the message holder u in a given step has knowledge of

1. the set of its local contacts.

2. the location of the target t.

The **expected delivery time**: expected number of steps taken by the algorithm to deliver the message over a network generated according to the model, from a source to a target chosen uniformly at random from the set of nodes.

How does the network model affect the expected delivery time of a decentralized algorithm?

Theorems

Theorem 1. *There is a decentralized algorithm* $\mathcal{A}$ *and a constant* α_2*, independent of* n*, so that when* $r = 2$ *and* $p = q = 1$*, the expected delivery time of* $\mathcal{A}$ *is at most* $\alpha_2(\log n)^2$*.*

Theorem 2. *(a) Let* $0 \leq r < 2$*. There is a constant* α_r*, depending on* p, q, r*, but independent of* n*, so that the expected delivery time of any decentralized algorithm is at least* $\alpha_r n^{(2-r)/3}$*.*

(b) Let $r > 2$*. There is a constant* α_r*, depending on* p, q, r*, but independent of* n*, so that the expected delivery time of any decentralized algorithm is at least* $\alpha_r n^{(r-2)/(r-1)}$*.*

Proof of Theorem 1

The obvious, but complicated, way to prove this theorem is to generate every possible graph, calculate the expected delivery time in that graph then weight that time by how likely that particular graph is to be generated. This approach isn't feasible so instead we use the principle of deferred decision. The graph is generated as the message is delivered.

The decentralized algorithm that achieves the bound Theorem 1 is: in each step, the current message-holder u chooses a contact that is as close (with respect to the lattice distance) to the target t as possible.

The probability that a node u chooses a particular node v as its long-range contact is $\frac{d(u,v)^{-2}}{\sum_{v \neq u} d(u,v)^{-2}}$ and we have

$$\sum_{v \neq u} d(u,v)^{-2} \leq \sum_{j=1}^{2n-2} (4j)(j^{-2})$$

$$= 4 \sum_{j=1}^{2n-2} j^{-1} \leq 4 + 4\ln(2n-2) \leq 4\ln(6n)$$

For $j > 0$, we say that the execution of the algo is in phase j when the lattice distance from the current node to t is greater than 2^j and at most 2^{j+1}. Note that in every phase the distance is to t is reduced by a factor of 2.

Suppose the algo is in phase j, $\log(\log n) \leq j \leq \log n$, at node u.

This phase will end when the message enters the set B_j of nodes within lattice distance 2^j of t. There are at least

$1 + \sum_{i=1}^{2^j} i > 2^{2j-1}$

nodes in B_j, each of which is within lattice distance $2^{j+1} + 2^j < 2^{j+2}$ of u.

The message enters B_j with probability at least

$\frac{2^{2j-1}}{4\ln(6n)2^{2j+4}} = \frac{1}{128\ln(6n)}$

This is the probability that any particular node is chosen as a long-range contact, times the number of nodes in B_j, divided by the number of nodes generating long-range edges.

Let X_j denote the total number of steps spent in phase j, $\log(\log n) \leq j \leq \log n$.

$$E[X_j] = \sum_{i=1}^{\infty} \Pr(X_j \geq i)$$

$$\leq \sum_{i=1}^{\infty} (1 - \frac{1}{128 \ln(6n)})^{i-1} = 128 \ln(6n)$$

If $0 \leq j \leq \log(\log n)$, then $E[X_j] \leq 128 \ln(6n)$.

Thus the total number of steps needed by the algo is

$$X = \sum_{j=0}^{\log n} X_j$$

Thus, $E[X] \leq (1 + \log n)(128 \ln(6n) \leq \alpha_2 (\log n)^2$ for a suitable α.