CS 683 Advanced Design and Analysis of Algorithms February 25, 2008

Lecture 16: Small World Graphs

Instructor: John Hopcroft *Scribe: Cristian Danescu Niculescu-Mizil, Myle Ott*

Small World Graphs

Local algorithm for $r = 2$

In the last lecture we discussed Small Word Graphs. We continue now with a proof that when $r = 2$, there exists a local algorithm to find a short path from a source vertex s, to a destination vertex t, in $O(\log n)$ steps. We reuse Kleinberg's [1] algorithm from Lecture 15.

Algorithm: At each step, select an edge that gets you closest to the destination.
Proof: In order to show that this algorithm takes $O(\log n)$ steps, we will divide it into phases. We will say that the algorithm is in phase j if the distance between the current vertex u, and the destination vertex t, is in the range $\left(2^j, 2^{j+1}\right]$ as shown in Figure 1. Thus, the expected running time is equal to the expected number of phases, which is $O(\log n)$, multiplied by the expected time spent in each phase.

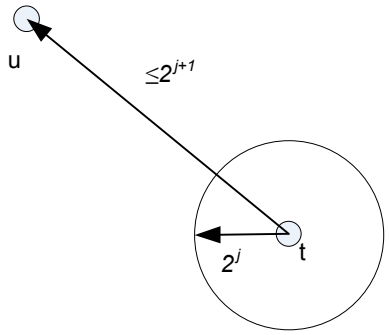

Figure 1: Escape from phase j into phase $j + 1$.

First recall that for small world graphs we have $P(u \to v) \propto d^{-r}(u, v)$ or, equivalently for $r = 2$:

$$P(u \to v) = \frac{d^{-2}(u, v)}{\sum_{w \neq u} d^{-2}(u, w)} \tag{1}$$

where the denominator is a normalizing factor.

Technical Lemma:

$$P(u \to v) \geq c \frac{d^{-2}(u, v)}{\ln n} \tag{2}$$

Proof: Given our definition of $P(u \to v)$ for small world graphs, we need to determine an upper bound on $\sum_{w \neq u} d^{-2}(u, w)$:

$$\begin{aligned}
\sum_{w \neq u} d^{-2}(u,w) &\leq \sum_{\text{distance } j} \frac{\#\text{ paths of distance } j}{j^2} \\
&\leq \sum_{j=1}^{2(n-1)} \frac{4(2j-1)}{j^2} \\
&\leq 8 \sum_{j=1}^{2(n-1)} \frac{1}{j} - 4 \sum_{j=1}^{2(n-1)} \frac{1}{j^2} \\
&\leq 8 \sum_{j=1}^{2(n-1)} \frac{1}{j} \\
&\leq \frac{1}{c} \ln n \;\leq\; c' \ln n \qquad (3)
\end{aligned}$$

In deriving the above, we made use of several facts. First, the maximum distance between two vertices in the lattice is $2(n-1)$ (the minimum distance is 1). Second, for any given distance j, there are at most $4(2j-1)$ paths[1] of distance j away from u (see Figure 2).

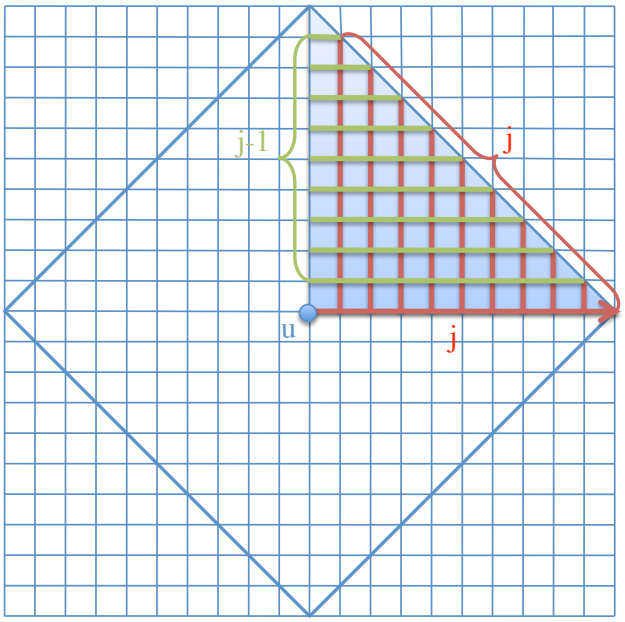

Figure 2: Given that there is a vertex at every grid crossing, the number of paths distance j away from vertex u is: $4 \times$ (the number of paths in the shaded triangle). There are j red paths achieved by going right some number of units, and then going up some number of units. Likewise, there are $j-1$ green paths achieved by going up some number of units and then going right some number of units. Thus, there are $4(2j-1)$ paths in total.

Lemma: The probability that there exists a long distance edge from u to some vertex in phase $j-1$ is greater than or equal to $\frac{c}{32 \ln n}$.

[1]This is a correction to the $4j$ value given in lecture.

Proof: Again consider Figure 1. We can see that:

$$\begin{aligned} Pr(u \to \text{some vertex in phase } j-1) &= (\#\text{ vertices } w \text{ in phase } j-1) \\ &\quad \times Pr(u \to \text{a specific vertex } w \text{ in phase } j-1) \\ &\geq 2^{(2j-1)} \times \frac{c}{\ln n \cdot \left[\max\limits_{w \in \text{phase } j-1} \{dist(u,w)\}\right]^2} \\ &\geq \frac{c}{\ln n} \cdot \frac{2^{(2j-1)}}{2^{(2j+4)}} \\ &\geq \frac{c}{32 \ln n} \end{aligned} \tag{4}$$

where,

$$\begin{aligned} (\#\text{ vertices } w \text{ in phase } j-1) &\geq \sum_{i=1}^{2^j} i \\ &\geq \frac{2^j(2^j+1)}{2} \\ &\geq 2^{2j-1} \\ \text{and,} \qquad \left[\max_{w \in \text{phase } j-1} \{dist(u,w)\}\right]^2 &= 2^{j+1} + 2^j \\ &< 2^{j+2} \end{aligned}$$

Claim: The expected time in each phase is $O(\log n)$.
Proof: Let X_j be a random variable representing the number of steps the algorithm makes in phase j. Then we have:

$$\begin{aligned} E(X_j) &= \sum_{i=0}^{\infty} i \cdot Pr(X_j = i) \\ &= Pr(X_j = 1) + 2Pr(X_j = 2) + 3Pr(X_j = 3) + \cdots \\ &= Pr(X_j = 1) + Pr(X_j = 2) + Pr(X_j = 3) + \cdots \\ &\qquad + Pr(X_j = 2) + Pr(X_j = 3) + \cdots \\ &\qquad + Pr(X_j = 3) + \cdots \\ &= Pr(X_j \geq 1) + Pr(X_j \geq 2) + Pr(X_j \geq 3) + \cdots \\ &= \sum_{i=1}^{\infty} Pr(X_j \geq i) \\ &= \sum_{i=1}^{\infty} \left(1 - \frac{c}{32 \ln n}\right)^{i-1} \\ &= \frac{1}{1 - \left(1 - \frac{c}{32 \ln n}\right)} = \frac{32 \ln n}{c} \end{aligned} \tag{5}$$

Thus, since there are $\log n$ phases, and we have shown that the algorithm is expected to spend $O(\log n)$ time in each phase, the expected time for the algorithm is $O(\log^2 n)$.

Non-existance of a local algorithm for $r < 2$

With an argument similar with the one employed in the last lecture for $r = 0$, we show here that it is impossible to use a local algorithm to find a short path in n^δ time, where $\delta < 1/2$.

Following Figure 3, consider the destination t and its neighborhood $N(t)$ of radius n^δ, which contains $n^{2\delta}$ vertices. Starting from the source s, we look into no more than n^δ edges to find a path that leads to some node in $N(t)$. Thus, there are at most $n^{3\delta}$ edges going into $N(t)$ that we could potentially inspect, each occurring with probability $\leq \frac{c}{n^{2-r}}$ (given without proof, as in lecture). Therefore, the probability that a local algorithm encounters a long distance edge ending in $N(t)$ is:

$$\begin{aligned} Pr(\text{long distance edge ending in } N(t)) &\leq \lim_{n\to\infty} n^{3\delta}\frac{c}{n^{2-r}} \\ &= 0 \quad \text{when} \quad \delta \leq \frac{2-r}{3} \end{aligned}$$

Since we know that we need such an edge in order to reach t from s (provided that they are picked uniformly at random), a local algorithm will never discover a short path from s to t.

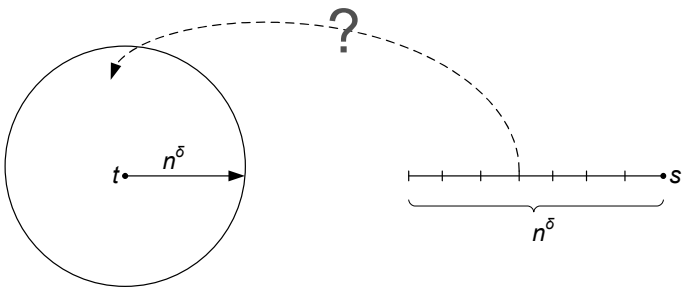

Figure 3: Can we find a long edge in n^δ steps?

Another explanation of the gap in random graphs

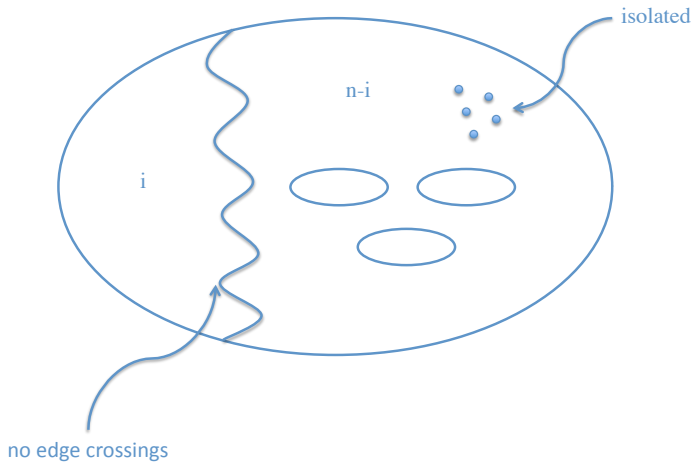

Figure 4: A component of size $\sim i$ and many small components

In order to explain "The Gap," Kozen shows that the probability of our graph having the structure

in Figure 4 goes to zero as $n \to \infty$, for $i \in [an, bn], a = \frac{1}{4}, b = \frac{3}{4}$.

$$\begin{aligned}
Pr(\text{Figure 4}) &= \sum_{i=an}^{bn} \binom{n}{i} (1-p)^{i(n-i)} \\
&\leq \sum_{i=0}^{n} \binom{n}{i} \left(1 - \frac{d}{n}\right)^{\frac{1}{4}n\frac{3}{4}n} \\
&\leq 2^n \left(1 - \frac{d}{n}\right)^{\frac{n}{d}\frac{d}{n}\frac{3n^2}{16}} \\
&\leq e^{n \ln 2} \cdot e^{-n\frac{3d}{16}} \\
&\leq e^{n(\ln 2 - \frac{3d}{16})} \quad , \quad \text{provided } \frac{3d}{16} \geq \ln 2 \\
&\leq e^{-cn} \quad , \quad \text{for some constant c,} \qquad (6) \\
&\qquad\qquad \text{where } p = \frac{d}{n}, \\
&\qquad\qquad \text{and } d \geq \frac{16 \ln 2}{3}
\end{aligned}$$

Thus, since the probability of (6) goes to 0 as $n \to \infty$, with high probability there will not exist a component of size i, for $i \in [\frac{1}{4}n, \frac{3}{4}n]$.

References

[1] Jon Kleinberg, *The Small-World Phenomenon: An Algorithmic Perspective*, SIGKDD'00: Proceedings of the 32nd ACM Symposium on Theory of Computing, 2000.