CS 591, Lectures 4+5
Graph Analytics
Boston University

Charalampos E. Tsourakakis

February 24th, 2020

Today's agenda

1. Navigation in a small-world
   - Model
   - Centralized vs. decentralized navigation algorithm
   - Kleinberg's result
   - Detailed proof of Kleinberg's elegant theorem

   End of random graph module

   Starting the space-efficient graph mining module

2. Space-efficient diameter estimation for massive graphs
   - Distinct elements estimation (Flajolet-Martin sketches)
   - ANF algorithm

Small-world phenomena

small worlds : graphs with short paths

- Stanley Milgram (1933-1984)
  "The man who shocked the world"
- obedience to authority (1963)
- small-World experiment (1967)

- we live in a small-world
- food for thought for class project: implications of small-world on spread of diseases (corona virus)?
  - E.g., Epidemics and percolation in small-world networks by Cristopher Moore, M. E. J. Newman

Small-world experiments

- letters were handed out to people in Nebraska to be sent to a target in Boston
- people were instructed to pass on the letters to someone they knew on first-name basis
- the letters that reached the destination (64 / 296) followed paths of length around 6
- Jon Kleinberg focused mathematically on the navigability of the small world using local information, and proved an elegant result we will go over today in great detail.
- Techniques useful for [HW1, Problem 4]

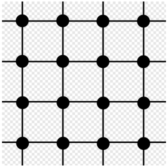
Navigation in a small world

Jon Kleinberg

how to find short paths using only local information?

- First, we will introduce Kleinberg's model.
- Remark. Results generalize to the d-dimensional grid.

Navigation in a small world - Kleinberg's model

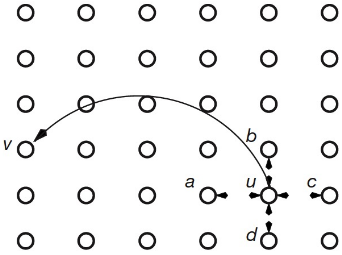


- Consider a $n \times n$ 2-dimensional lattice/grid.
  - Each node has either 4 (interior), 3 (boundary) or 2 (corners) neighbors.
  - Lattice distance: Consider two points $x = (x_1, x_2), y = (y_1, y_2)$. Their distance is $r(x, y) = |x - y|_1 = |x_1 - y_1| + |x_2 - y_2|$.
- Each node x creates randomly a shortcut edge.
  - How?

Navigation in a small world - Kleinberg's model

$r(u, v)$: shortest path distance using only original grid edges

directed graph model, parameter $s > 0$:



(source [Kleinberg, 2000])

- for each vertex v we add an extra link (v, u) where u is chosen with probability proportional to $d(v, u)^{-s}$

Navigation in a small world – Centralized vs. decentralized

- Both a **centralized** and **decentralized** algorithm have access to source u, target v, and want to find a short path from u to v.
- A **centralized** algorithm has access to full network, i.e., knows all shortcuts.
  - E.g.,the shortest path can be found using Dijkstra's algorithm.
- A **decentralized** algorithm is "local":
  - When at a certain node, it only has access to **previously** visited nodes and their connections (lattice and long range).
  - Even if short paths between u, v exist may exist it may not be able to detect them

Navigation in a small world – Results

High level description of Kleinberg's results:

- $s = 0$: random edges, independent of distance
- as s increases the length of the long distance edges decreases in expectation

results

1. $s < 2$: shortcuts have long-range but they are "spread out".
   - No navigability!
2. $s = 2$: there are short paths and a simple greedy algorithm finds them
3. $s > 2$: encountering "a long shortcut is not likely"

Case I: $s = 2$

Theorem

Consider the greedy routing algorithm that works as follows:

- *At each step, we choose to route the message to the neighbor of the current node that is closest to target (ties broken uar).*

For any source/target pair (u, v) the expected number of steps is $O(\log^2 n)$.

Proof on blackboard.

Case II: $s > 2$

Theorem

Fix source u, target v with distance $r(u,v) > \frac{n}{4}$. Then for any decentralized routing algorithm, the expected number of steps for message delivery is $\Omega(n^{\frac{s-2}{s-1}})$.

Proof on blackboard.

Case III: $s < 2$

Theorem

Fix source u, target v with distance $r(u, v) > \frac{n}{4}$. Then for any *decentralized routing algorithm, the expected number of steps for message delivery is $\Omega(n^{\frac{2-s}{3}})$.*

Proof on blackboard.

Today's problem: Distinct Elements

- Given a stream of integers $< x_1, \ldots, x_m >$ where $x_i \in [U] := \{1, 2, \ldots, u\}$, output the number n of distinct elements seen.

- Example: There exist 5 distinct elements in the stream $< 3, 3, 1986, 1, 6, 12, 1, 12, 6, 1, 3 >$, i.e., $n = 5$.

- The number of distinct elements of a stream is also known as its (F_0) moment.

Claim: To solve the distinct elements problem (F_0) exactly we need at least $\min(\{m \log u, u\})$ space.

$\mathbb{E}\left[\min(X_1, \ldots, X_n)\right] = \frac{1}{n+1}$

- $X_i \in U[0, 1]$ for $i \in [n]$
- $Z = \min(X_1, \ldots, X_n)$

$$\begin{aligned}\mathbb{E}\left[Z\right] &= \int_0^1 \mathbf{Pr}\left[Z > t\right] dt = \int_0^1 \mathbf{Pr}\left[X_1 > t\right]^n \\ &= \int_0^1 (1-t)^n dt = \frac{1}{n+1}.\end{aligned}$$

A slick proof follows ...

$\mathbb{E}[\min(X_1, \ldots, X_n)] = \frac{1}{n+1}$

- $X_{n+1} \in U[0, 1]$
- What is $\mathbf{Pr}[X_{n+1} < \min(X_1, \ldots, X_n)]$ equal to?

1 By symmetry to $\frac{1}{n+1}$

2 On the other hand by definition of uniform distribution, it is equal to $\mathbb{E}[\min(X_1, \ldots, X_n)]$.
QED

Hashing!

Suppose that we have access to a random hash function $h : [u] \to [0, 1]$.

FM method (Flajolet-Martin)

- We initialize $X \leftarrow +\infty$.
- When x_i arrives, we use h to hash it to $h(x)$
- If $h(x) < X$ we set $X \leftarrow h(x)$
- At the end of the stream, $X = \min_{x \in \text{stream}} h(x)$
- Output $1/X - 1$

Issues

- To store h we need $\Omega(u)$ space
- Floating-Point Arithmetic

FM+

Idea: Average together multiple estimates from the idealized algorithm FM.

1. Instantiate $q = \frac{1}{\varepsilon^2 \eta}$ FMs independently
2. Let X_i come from FM_i.
3. Output $1/Z - 1$, where $Z = \frac{1}{q} \sum_i X_i$.

To analyze FM+ we need to upper bound the variance of each X_i, and apply Chebyshev's inequality.

FM+ Analysis

$$\mathbb{E}\left[X^2\right] = \int_0^1 \mathbf{Pr}\left[X^2 > t\right] dt = \int_0^1 (\mathbf{Pr}\left[X_1^2 > t\right])^n dt = ..$$
$$= \frac{2}{(n+1)(n+2)}.$$

Therefore, the variance $\mathbb{V}ar\left[X\right]$ is equal to

$$\mathbb{V}ar\left[X\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}\left[X\right]^2 = \frac{n}{(n+1)^2(n+2)} < \frac{1}{(n+1)^2}$$

FM+ Analysis

Theorem

$$\mathbf{Pr}\left[|Z - \frac{1}{n+1}| > \frac{\varepsilon}{n+1}\right] < \eta.$$

Proof.

We apply Chebyshev's inequality

$$P(|Z - \frac{1}{n+1}| > \frac{\varepsilon}{n+1}) < \frac{(n+1)^2}{\varepsilon^2} \frac{1}{q(n+1)^2} = \eta$$

□

Notice that we care about the concentration of $\frac{1}{Z}$, not Z.

FM+ Analysis

Theorem

$\mathbf{Pr}\left[|(\frac{1}{Z}-1)-n| > O(\varepsilon)n\right] < \eta$

Proof sketch: We use Taylor expansion as follows:

$$\frac{1}{(1\pm\varepsilon)\frac{1}{n+1}}-1=(1\pm O(\varepsilon))(n+1)-1=(1\pm O(\varepsilon))n\pm O(\varepsilon)$$

Median Boosting Trick: FM++

We use FM+ as a blackbox. We take multiple estimates of it, and we take the median.

1. Instantiate $s = \lceil 36 \ln(2/\delta) \rceil$ independent copies of FM+ with $\eta = 1/3$.
2. Output the median $\widehat{n}$ of $\{1/Z_j - 1\}_{j=1}^{s}$ where Z_j is from the jth copy of FM+.

FM++ Analysis

Theorem: $\mathbf{Pr}\left[|\widehat{n} - n| > \varepsilon n\right] < \delta$.
Proof:
Let

$$Y_j = \begin{cases} 1 & \text{if } |(1/Z_j - 1) - n| > \varepsilon n \\ 0 & \text{else} \end{cases}.$$

using Chernoff

$$\mathbf{Pr}\left[\sum Y_j > s/2\right] = \mathbf{Pr}\left[\sum Y_j - s/3 > s/6\right] =$$
$$\mathbf{Pr}\left[\sum Y_j - \mathbb{E}\sum Y_j > \frac{1}{2}\mathbb{E}\sum Y_j\right] < e^{-\frac{(\frac{1}{2})^2 s/3}{3}}$$
$$< \delta$$

2-wise independent family

Reminder from Algorithms' prereq: We can construct a 2-wise independent family as follows.

- p is prime
- $a \neq 0, b$ chosen uar from $[p]$
- The hash of x is

$$h(x) = ax + b \mod p,$$

High level idea - Diameter and F_0 moment

- Assume that for each vertex v in the graph, we maintain the number of neighbors reachable from v within h hops.
- As h increases, the number of neighbors increases until it stabilzes.
- The diameter is h where the number of neighbors within $h+1$ does not increase for every node.

This suggest the following idea:

1. For each vertex i, create a set S_i and initialize it by adding i to it.
2. For each vertex i, continue updating S_i by adding 1,2,3,...-step neighbors of i to S_i. When the size of S_i stabilizes for first time, then the vertex i reached its radius. Iterate until all vertices reach their radii.

High level idea - Diameter and F_0 moment

- Caveat: too much space required!
  - n vertices
  - Each vertex requires $\Omega(n)$ space.
  - Total space requirement is $\Omega(n^2)$.
- **Prohibitive** for large-scale graphs
- **Key idea:** Use space-efficient sketches to estimate distinct elements.
  - The ANF algorithm used FM sketches [Palmer et al., 2002]
  - More recently, Boldi-Rosa-Vigna used Hyperloglog counters [Boldi et al., 2011]
- These tools have been used to analyze the Web graph [Kang et al., 2011] and the Facebook graph [Backstrom et al., 2011].

ANF algorithm

- Implementation of FM sketches
  - We maintain a bitstring $BITMAP[0 \dots L-1]$ of length L which encodes the set.
  - For each item we add, we do the following:
    1. Pick an $index \in [0 \dots L-1]$ with probability $1/2^{index+1}$.
    2. Set $BITMAP[index]$ to 1.
  - Let R denote the index of the leftmost '0' bit in $BITMAP$.
  - The unbiased estimate of the size of the set is given by

$$\frac{1}{0.77351} 2^R.$$

ANF algorithm

- We maintain K Flajolet-Martin (FM) bitstrings $b(h, i)$ for each vertex i and the current hop number h.
- $b(h, i)$ encodes the number of vertices reachable from vertex i within h hops
- $b(h, i)$ are iteratively updated until the bitstrings of all vertices stabilize.
- At the h-th iteration, each vertex receives the bitstrings of its neighboring vertices, and updates its own bitstrings $b(h-1, i)$ handed over from the previous iteration:

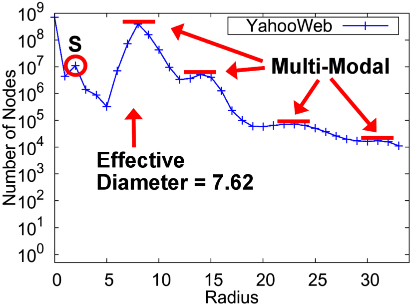
$$b(h, i) = b(h-1, i) \text{ BIT-OR } \{b(h-1, j) | (i, j) \in E\}$$

ANF algorithm

- Let $N(h, i)$ be the number of vertices within h hops from the vertex i.
- $N(h, i)$ is estimated from the K bitstrings by

$$N(h, i) = \frac{1}{0.77351} 2^{\frac{1}{K} \sum_{l=1}^{K} b_l(i)}$$

where $b_l(i)$ is the position of leftmost '0' bit of the l^{th} bitstring of vertex i.



Readings

- Kleinberg's algorithmic perspective on the small world phenomenon [Kleinberg, 2000] and Chapter 20 – Easley-Kleinberg book
- The ANF paper and the subsequent improvements (Hyperloglog counters and HyperANF) [Palmer et al., 2002, Boldi et al., 2011]
- Applied papers (Findings in HADI+Four degrees of separation) [Kang et al., 2011, Backstrom et al., 2011]

references I

Backstrom, L., Boldi, P., Rosa, M., Ugander, J., and Vigna, S. (2011).
Four degrees of separation.
CoRR, abs/1111.4570.

Boldi, P., Rosa, M., and Vigna, S. (2011).
HyperANF: approximating the neighborhood function of very large graphs on a budget.
In *WWW*.

Kang, U., Tsourakakis, C. E., Appel, A. P., Faloutsos, C., and Leskovec, J. (2011).
HADI: Mining radii of large graphs.
ACM TKDD, 5.

references II

Kleinberg, J. M. (2000).
Navigation in a small world.
Nature, 406(6798):845–845.

Palmer, C. R., Gibbons, P. B., and Faloutsos, C. (2002).
ANF: a fast and scalable tool for data mining in massive graphs.
In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 81–90, New York, NY, USA. ACM Press.