COMS E6998-9: Algorithmic Techniques for Massive Data — Feb 26, 2019

Lecture 11 : Connectivity, distance, & triangle-counting in graphs

Instructor: *Alex Andoni* — Scribe: *Yiguang Zhang*

1 Motivation

- graph on hard drive.
- one time scan over the graph is much faster than in random process.
- most graph algorithms are very non-local.
- keep small working memory, where we allow random access. This is equivalent to the space of streaming models.

2 Common Graph Problems

- **Connectivity**
- **Distance**
- Pagerank
- Graph partition
- **Triangle counting (measure of the clusterability of graphs)**

3 Connectivity

Goal: use space $\sim n = \#$ nodes $<< m = \#$ edges to check if an undirected graph is connected.

Idea: keep a spanning tree/forest.

Algorithm:

- init $H = \emptyset$.
- Add edge (i, j) to H if and only if no path $i \leftrightarrow j$ in H.

Correctness: by construction of H, it it is a spanning tree/forest.

Space: $|H| \leq n - 1$ edges. The space needed is $O(n)$ words.

4 Distance

In this section, we consider only undirected, unweighted graphs. The method can be applied to weighted graphs, but for directed graphs, we usually need much more space.

Theorem 4.1 Given a graph G and two nodes i and j, it takes $\Omega(n)$ space to calculate the exact distance between i and j.

Approximation: $\alpha > 1$, where α is an odd integer.

Algorithm

- init $H = \emptyset$.
- add (i, j) to H if and only if $dist_H(i, j) > \alpha$.
- output $dist_H(i^*, j^*)$.

Claim 4.2:

$$dist_G(i^*, j^*) \leq dist_H(i^*, j^*) \leq \alpha \cdot dist_G(i^*, j^*).$$

Proof. By construction of H, for all path $i^* \to j^*$, there exists alternative path in H of length $\leq \alpha\times$ more. □

Space We will adapt the following theorem:
Theorem 4.2 [Bollobas] if all cycles of H have length $n \geq d + 2$, then $|H| \leq O(n^{1+\frac{2}{\alpha+1}})$.

Note that in our H, all cycles have length $\geq (\alpha + 1) + 1 = \alpha + 2$. Therefore, we need $O(n^{1+\frac{2}{\alpha+1}})$. Now let's prove that Theorem 4.2 holds for all d-regular graphs:

Proof. For a simplified case, let us assume all nodes have degree d.

- Suppose $\alpha = 2k - 1$
- We fix a vertex v and get a BFS tree that's rooted at v.
- Note that at depth k of the BFS tree, all nodes differ (otherwise the graph would not be d-regular). Therefore,

$$\begin{aligned} d^k &\leq n && (1) \\ d &\leq n^{1/k} && (2) \\ m &\leq n^{1+1/k} && (3) \\ &= n^{1+\frac{2}{1+\alpha}} && (4) \end{aligned}$$

□

Theorem 4.3 for all undirected graph G, for any integer $\alpha = 2k - 1$, where $k \geq 2$, there exists a graph H such that

1. $|H| \leq O(kn^{1+1/k})$
2. $dist_G(i,j) \leq dist_H(i,j) \leq \alpha \cdot dist_G(i,j)$.
3. there exists a data structure with space $O(kn^{1+1/k})$, and the distance query can be processed in $O(k)$ time.

5 Triangle counting

Let T = number of triangles in the graph
Physical motivation: to answer some questions like "How often do two friends of a person know each other?"

Define this fraction as

$$F = \frac{3T}{\sum_v \binom{deg(v)}{2}} \in [0, 1].$$

- Denominator
  - It is possible to measure the denominator by just counting the degrees of vertices
  - $O(n)$ space required to do this
- Numerator T
  - Measuring the numerator is harder
  - It is not possible to distinguish $T = 0$ from $T = 1$ in $<< m$ space
  - Suppose we have a lower bound $t \leq T$
  - We provide an $(1 \pm \epsilon)$ approximation in the following subsection.

5.1 Triangle counting : Approach

Define a vector x which has a coordinate x_S for each subset S of three nodes. The value of this coordinate is

- x_S = number of edges among vertices in S
- T = number of coordinates in x that have value of 3

We had earlier defined frequencies as

- $F_p = \sum_S x_S^p$

Claim: $T = F_0 - 1.5F_1 + 0.5F_2$
This is equivalent to writing
$\sum_S \chi[X_S \neq 0] - 1.5 \sum_S X_S^1 + 0.5 \sum_S X_S^2 = \sum_S \chi[X_S = 3]$

Proof. Fix S,

- $X_S = 0$ contribute 0 to both LHS and RHS

- $X_S = 1$, which means there is exactly one edge. This contributes 0 to RHS.
  - LHS evaluates to $1 - 1.5 * 1 + 0.5 * 1^2 = 0$
- $X_S = 2$ contribute 0 to both LHS and RHS
  - LHS evaluates to $1 - 1.5 * 2 + 0.5 * 2^2 = 0$
- $X_3 = 3$ contributes 1 to RHS
  - LHS evaluates $1 - 1.5 * 3 + 0.5 * 3^2 = 1$

  We can generate such a formula because of polynomial interpolation.
  - We need a polynomial $f(X_S)$ that evaluates to 0 on $\{0, 1, 2\}$ and evaluates to 1 on $\{3\}$
  - Use polynomial interpolation!
  - We ideally need a polynomial of degree 3 but we get one degree of freedom from F_0 so 2 is enough.

$\square$

Goal: Estimate F_0, F_1, F_2 of the (implicit) vector x up to $(1 + \gamma)$-factor approximation.

Algorithm

- Let $\hat{F}_0, \hat{F}_1, \hat{F}_2$ be $1 + \gamma$ estimates
- Stream the edges to generate updates for X_S
  - For each edge $e = (i, j)$
  - Generate S that contain these two nodes
  - For each $S = \{i, j, k\}$, set $X_S = X_S + 1$
- Estimate $\hat{T} = \hat{F}_0 - 1.5 * \hat{F}_1 + 0.5 * \hat{F}_2$

Analysis (we did not finish this part in class)

- Note that $\hat{T}$ is not an $(1 \pm \gamma)$ estimator, even if each of the terms is a $(1 \pm \gamma)$-factor approximation (in particular, due to the minus sign – F_0 and F_1 can be large, while T is small). Hence we use the following guarantees on additive approximation:
  - $|\hat{F}_0 - F_0| < \gamma F_0$
  - $|\hat{F}_1 - F_1| < \gamma F_1 \leq 3\gamma F_0$
  - $|\hat{F}_2 - F_2| < \gamma F_2 \leq 9\gamma F_0$
- Using the above, we get error in $\hat{T} = O(\gamma F_0) = O(\gamma mn)$
- Therefore we can set $\gamma = \frac{O(t)}{\epsilon mn}$ for a $\pm\epsilon t$ additive error

- Total space required is

$$O(\gamma^{-2} \log n) = O\left(\left(\frac{mn}{\epsilon t}\right)^2 \log n\right)$$