COMS E6998-9: Algorithmic Techniques for Massive Data — Sep 24, 2015

Lecture 6 : Counting triangles, Dynamic graphs & sampling

Instructor: *Alex Andoni* — Scribe: *Patanjali Vakhulabharanam*

Plan

- Counting triangles
- Streaming for dynamic graphs

1 Streaming for graphs(recap)

Consider a graph with

- n vertices
- m edges

which is represented as a stream of list of edges, stored somewhere like on a hard drive. We can sequentially access this data to generate a stream. Number of edges m could be $O(n^2)$ If we have a limited working memory and are trying to process the edge stream, we would like to get an algorithm that uses space

- $O(n)$
- $O(n \log n)$ which is still much better than $O(n^2)$

Acheiving $<< n$ is usually not possible.

1.1 Problems

1. Connectivity
   - Exact in $O(n)$ space
2. Distances
   - α (odd) approximation in $O\left(n^{1+\frac{2}{\alpha+1}}\right)$
3. Count # of triangles

2 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

$$F = \frac{T}{3\sum_v \binom{deg(v)}{2}}$$

$F \in [0, 1]$

- 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 $t \leq T$

2.1 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

- $X_S = 0$ contribute 0 to both LHS and RHS
- $X_S = 1$ contribute 0 to both LHS and 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.

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 $X_S = X_S + 1$
  - This is the rule $(S, +1)$
- Estimate $\hat{T} = \hat{F}_0 - 1.5 * \hat{F}_1 + 0.5 * \hat{F}_2$
- The errors for each of the terms cannot be directly added. We use the following inequalities
  - $|\hat{F}_0 - F_0| < \gamma F_0$
  - $|\hat{F}_1 - F_1| < \gamma F_1 \leq 3\gamma F_0$
  - $|\hat{F}_2 - F_2| < \gamma F_2 \leq 9\gamma F_0$
- 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
$$O(\gamma^{-2} \log n) = O\left(\left(\frac{mn}{\epsilon t}\right)^2 \log n\right)$$

Algorithm 2 Let us consider an even simpler algorithm that the previous one

- Pick a few random S_i for $i \in [k]$of 3 nodes
- Compute X_{S_i} for $i \in [k]$

- Let c be the number of i such that $X_{S_i} = 3$
- Estimate $R = \frac{M}{k} * c$ where $M = \binom{n}{3}$
- Mean of R is

$$\begin{aligned}
E[R] &= E[\frac{M}{k} * c] \\
&= E[\frac{M}{k} * \sum_S (\chi[X_S = 3] * \chi[S \ is \ sampled])] \\
&= \frac{M}{k} * \frac{k}{M} * T \\
&= T
\end{aligned}$$

- Variance of R is

$$\begin{aligned}
Var[R] &= \sum_S Var[\frac{M}{k} * \chi[X_S = 3] * \chi[S \ is \ sampled]] \\
&\leq \sum_{S|X_S=3} \left(\frac{M}{k}\right)^2 * Probability(S \ is \ sampled) \\
&= \frac{M^2}{k^2} * \frac{k}{M} * T \\
&= \frac{TM}{k}
\end{aligned}$$

- Using Chebyshev's inequality, we get $|R - T| \leq O\left(\sqrt{(\frac{MT}{K})}\right)$
- We need $k = \frac{O(1)}{\epsilon^2} * \frac{M}{t} = O\left(\frac{1}{\epsilon^2} * \frac{n^3}{t}\right)$, since $M = O(n^3)$

Algorithm 2+

- The previous algorithm can be improved by choosing S more selectively rather than randomly.
- Pick only those S for which $X_S \geq 1$
- The size of this set will be $M' << M$
- Using Chebyshev's inequality on this, we get
  - $|R - T| \leq O\left(\sqrt{(\frac{M'T}{K})}\right)$
- Using this and $M' = O(mn)$, we get
  - $k = \frac{O(1)}{\epsilon^2} * \frac{M'}{t} = O\left(\frac{1}{\epsilon^2} * \frac{mn}{t}\right)$

3 Sampling in graphs

- Setting 1
  - Updates are only positive
  - Not linear
- Setting 2
  - General streaming : Also include negative updates
  - This is motivated by dynamic graphs were connections can get added as well as deleted

Dynamic grahps - Streams can contain both insertions and deletions of edges. There are several use cases for such graphs

- Use 1 : Log file of updates to the graph
  - A graph of a social network can have people "unfriending"
  - A graph of webpage hyperlinks can have links being added as well as deleted, etc
- Use 2 : Graph is distributed over a number of computers
  - We will then want linear sketches
  - In general dynamic streams and linear sketches go together
- Use 3 : If the algorithm is time efficient, it can also be considered a data structure. This makes it interesting to areas that are beyond just algorithms.

4 Revisiting connectivity

- Can we do connectivity in dynamic graphs using the algorithm from previous lecture?
- No

  Theorem 1. *We can check s-t connectivity in dynamic graphs with $O(nlog^5 n)$ space (with 90% success probability)*

Approach : Use sampling in (dynamic) graphs We wll first look at a sub problem - dynamic sampling

5 Dynamic sampling

Problem

- General updates to a vector $x \in \{-1, 0, 1\}^n$
  - This will also work for a general x
- Goal : Output i with probability $\frac{|x_i|}{\sum_j |x_j|}$

- Does standard sampling work?
  - No : For instance consider this. After putting $x_i = 1$ for $n/2$ coordinates, add 1 more and delete the first $n/2$
- Let $A = \{isuchthatx_i \neq 0\}$
- Intuition
  - Suppose $|A| = 10$.
    * How do we sample i with only non zero x_i
    * Notice that each of the x_i which are non zero are $\frac{1}{10}$ heavy hitters
    * Therefore using CountSketch, we can recover all of them
    * $O(\log n)$ space is required for this
  - Suppose $A = n/10$
    * Downsample first - Pick a random subset $D \subset [n]$ of size $|D| \approx 100$
    * Focus on a substream $i \in D$ only and ignore the rest
    * $E[|A \cap D|] = 10$
    * Use CountSketch on the downsampled stream
  - In general, prepare for all levels