0368.416701 Sublinear Time Algorithms. Fall 2015	9.11.15

Lecture 4

Instructor: Ronitt Rubinfeld Esty Kelman, Gal Hyams, Uri Meir, Tom Jurgenson

Plan for today

1. More on probability testing.
2. Estimate the number of connected components in a graph.

1 Testing for monotonicity of a distribution

Def: distribution p over domain $[n]$ is "monotone decreasing" if

$$\forall i \in [n-1] : p(i) \geq p(i+1)$$

Goal: design an algorithm such that:

1. if p is monotone decreasing output PASS (with probability $\geq \frac{3}{4}$)
2. if p is ϵ-far from monotone decreasing output FAIL (with probability $\geq \frac{3}{4}$)

A useful tool - Birge Decomposition: Given any monotone decreasing distribution q and ϵ, we decompose the domain $[n]$ into $l = \Theta(\frac{\log \epsilon n}{\epsilon}) \approx \Theta(\frac{\log n}{\epsilon})$ intervals $I_1^\epsilon, I_2^\epsilon, ...I_l^\epsilon$ such that:

$$|I_{k+1}^\epsilon| = \lceil (1+\epsilon/2) \cdot |I_k^\epsilon| \rceil$$

Note: for notation purposes, we disregard ϵ and simply denote I_k.

Define $\tilde{q}_\epsilon$ - "the flattened distribution": $\tilde{q}_\epsilon$ "flattens" each part $q(I_j)$ of the partition, by distributing uniformly on it's values. Namely:

$$\forall j \in [l], \forall i \in I_j : \tilde{q}_\epsilon(i) = \frac{q(I_j)}{|I_j|}$$

Making the original distribution into a "staircase" distribution, where each part of the partition is one stair, and each part keeps it's weight as the original weight it had in q.

Important Theorem:

1. If distribution q is monotone decreasing then $\|\tilde{q}_\epsilon - q\|_1 < \epsilon$.
2. If distribution q is ϵ-close to any monotone decreasing (with respect to l_1 distance) then $\|\tilde{q}_\epsilon - q\|_1 < O(\epsilon)$.

Algorithm 1 Testing decreasing monotonicity

1: For each part I_j in the partition: test whether $q_{|I_j}$ is close to uniform. If not, output FAIL

2: $w_j \leftarrow$ estimate weights of each partition I_j.

3: Use LP to verify that that w is close to monotone

3. If distribution q is monotone decreasing then for each part I_j in the partition, we have :$q_{|I_j}$ is close to uniform.

Sample analysis: The number of samples required for the above algorithm is
$\Omega(\frac{\sum \sqrt{I_j}}{\epsilon^2}) \cdot \Theta(\frac{\log n}{\epsilon}) = \Omega(\frac{\sqrt{n} \cdot \log n}{\epsilon^3})$.
The first term is for testing uniformity in each interval, and the second term is the number of parts in our partition.

Notes:

1. If at any interval the number of samples is too small approximate by 0.
2. Normally, step 2 is hard, but under the notion that the number of partitions is $\Theta(\frac{\log n}{\epsilon})$, the LP is easily solvable.
3. The correctness of this algorithm is also derived from the following observation:
For 2 probability distribution p, q over the same partitions $I_1^\epsilon, I_2^\epsilon, ...I_l^\epsilon$, if the conditional distributions hold: $\forall I_j : \|p_{|I_j} - q_{|I_j}\|_1 < \epsilon$, then we get: $\forall I_j : \|p - q\|_1 < \epsilon$.
4. The first step of Algorithm 1 is checking closeness of each part $q_{|I_j}$ to a uniform distribution. The algorithm for that was shown in the previous lecture, and generally, it is not 'tolerant'. Meaning, it might output FAIL for distributions that are ϵ-close to uniform.
Luckily, for our possible inputs of $q_{|I_j}$, it can be made tolerant enough to keep the correctness of Algorithm 1.

And now for something completely different.

2 Estimate the number of connected components in a graph

Given an undirected graph G(V,E), represented as an adjacency-list, and a (relatively small) number d let us define: $n := |V|$, $m := |E|$. we only consider sparse graphs, where d is significantly smaller than n, and $max_{v \in V} degree(v) \leq d$.
Generally, sub-linear algorithms over graphs are considered sub-linear time in m, but since it is possible that m=0... particularly in this problem, we will consider the sample complexity to be sub-linear in $(m + n)$.
In this algorithm, we will see that the sample complexity will depend on d, and not on n or m.

Definition 2.1 *A connected component of an undirected graph is a sub-graph in which any two vertices are connected to each other by a path.*

We will see an algorithm for estimating the number of connected components in an undirected graph G. our algorithm will return a value y s.t.

$$C - \epsilon n \leq y \leq C + \epsilon n$$

where $C =\#$ connected components. i.e. we get a bound on the distance: $|y - C| \leq \epsilon n$

Notes:

1. There is a lower bound on the sample complexity for this algorithm in terms of ϵ and d.

2.ϵn might be very big as to C. Meaning if G is a very big graph (big n), that has little connected components (small C), we might get a big error, related to C.

We note that this does not concern us, since we are interested in the cases where C is big. Namely, we have many connected components in the graph.

We begin with some definitions and observations in order to see how we build such algorithm:

Definition 2.2 *Let v be a vertex in V. We define n_v as the number of vertices in the connected component to which v belongs. Namely: $\forall v \in V$, let $n_v := \#\{u \in V / \exists \text{path between } u \text{ and } v\}$*

Observation 1: $\forall$ connected component $A \subseteq V$:

$$\Sigma_{v \in A} \frac{1}{n_v} = \Sigma_{v \in A} \frac{1}{|A|} = \frac{|A|}{|A|} = 1 \Rightarrow \Sigma_{v \in V} \frac{1}{n_v} = C$$

(where the rightmost equation comes from the fact that we get 1 over the summation on each and every connoected component$A$.)

Allegedly: We need $n^2 d$ steps to precisely calculate C. We will now show an approximation that runs in sub-linear time. first, we approximate n_v and then we approximate the summation itself. We will show that we can estimate $\Sigma_{v \in V} \frac{1}{n_v}$ with a small amount of samples, using the standard Chernoff bound.

Recall that d is greater than the maximum degree in G. Let us consider d as a constant in the input, $d << n$.

since the graph is represented as an adjacency list, iterating over all neighbours of a given vertex takes at most d steps. We will estimate $\Sigma_{v \in V} \frac{1}{n_v}$ in two steps:

1) estimating $\frac{1}{n_v}$

2) estimating the sum of our values using Chernoff bound.

Step 1: Estimating $\frac{1}{n_v}$

Definition 2.3 *we define: $\hat{n}_v := min\{n_v, \frac{2}{\epsilon}\}$*

We notice that it means: $\frac{1}{\hat{n}_v} = max\{\frac{1}{n_v}, \frac{\epsilon}{2}\}$

we can assume ϵ to be a significantly small number. Namely: $\epsilon << 1$, Hence, $\frac{2}{\epsilon} >> 1$.

This way, every vertex that belongs to a small connected component, will satisfy: $\hat{n}_v = n_v$.

The vertices that belong to large connected component, we can "round down", since the fraction $\frac{1}{n_v}$ will have a small affect on our summation.

Definition 2.4 *Let us define $\hat{C}$ as follows:* $\hat{C} = \Sigma_{v\in V}(\frac{1}{\hat{n}_v})$

Lemma 1:

$$\forall v\, |\frac{1}{\hat{n}_v} - \frac{1}{n_v}| \leq \frac{\epsilon}{2}$$

Proof: There are 2 possible cases:

1) if $n_{v\leq \frac{2}{\epsilon}}$, then $\hat{n}_v = n_v \Rightarrow |\frac{1}{\hat{n}_v} - \frac{1}{n_v}| = 0$

2) else: we have $n_v > \frac{\epsilon}{2}$.

Therefore: and then $\hat{n}_v = \frac{2}{\epsilon} \Rightarrow \frac{1}{\hat{n}_v} = \frac{\epsilon}{2}$, $\frac{1}{n_v} < \frac{\epsilon}{2} \Rightarrow |\frac{1}{\hat{n}_v} - \frac{1}{n_v}| = |\frac{\epsilon}{2} - \frac{1}{n_v}| \leq \frac{\epsilon}{2}$

The last inequality stands because $\frac{1}{n_v}$ is a positive number.

Now we will show that: $|\hat{C} - C| \leq |\Sigma_{v\in V}(\frac{1}{\hat{n}_v}) - \Sigma_{v\in V}(\frac{1}{n_v})| \leq \Sigma_{v\in V}|\frac{1}{\hat{n}_v} - \frac{1}{n_v}| \leq n \cdot \frac{\epsilon}{2} = \frac{\epsilon n}{2}$

where the second inequality comes from pairwise triangle inequality, and the third is true because $|V| = n$.

And now we have the next consequence:

Corollary 1:

$$|\hat{C} - C| \leq \frac{\epsilon n}{2}$$

Note that we have constructed the estimation of $\hat{n}_v$ s.t the estimate of $|\hat{C} - C|$ can only have half of the error range we had. This gives us room for some additive error in step 2 as well.

(Good question: given i , how can we choose random neighbours? can we check if j is neighbour? Answer: neighbour: running on i's adjacency list $O(d)$ for a neighbour (where d is the bound on the degree in G). check if j is a neighbour: again, running on the list. also $O(d)$)

Now we would like to calculate $\hat{n}_v$. How will we do this, and how long will it take?

Algorithm 2 calculating $\hat{n}_v$

1: We run a BFS until visiting whole connected component of v or until we see $\frac{2}{\epsilon}$ new nodes.

2: we output the number of nodes we visited during that process.

we note that the nmber of visited nodes is $= (min\{n_v, \frac{2}{\epsilon}\})$.

Complexity: This is bounded by $\frac{d*2}{\epsilon}$. so it's $O(\frac{d}{\epsilon})$, since every step of the BFS has time complexity of at most d, and there are at most $\frac{2}{\epsilon}$ steps of BFS.

So, We can calculate $\frac{1}{\hat{n}_v}$ in $O(\frac{d}{\epsilon})$ time for any vertex v.

Step 2: Estimating $\hat{C}$.

We start off by describing the algorithm for that calculation

where the choosing of $r = \frac{b}{\epsilon^3}$, depends on b, which is a constant that we will choose later on, using Chernoff bound. We will also see (at the proof of Theorem 1), that with high enough probability - that number of samples will suffice.

We notice that the estimation of $\hat{C}$ is adding us another place for error, since we only estimate it by taking an average over r samples and multiplying it by n.

Algorithm 3 estimating $\hat{C}$

1: We set $r := \frac{b}{\epsilon^3}$

2: We take r samples of $\hat{n}_v$.

3: choose $U = \{u_1, ..., u_r\}$ random nodes, uniformly.

4: $\forall u_i \in U$ compute $\hat{n}_{u_i}$, using Algorithm 2.

5: Sum and output: $\widetilde{C} = n \cdot \frac{1}{r}(\Sigma_{u_i \in U} \frac{1}{\hat{n}_{u_i}})$.

we will prove later on that most of the times, that estimation is good enough.

A possible problem: summing via the averages could create very rough estimation when dealing with samples that have big variance. for example:

Using that method for (1,2,2,3,4,4,3,2,1,4) will give us a good estimation.

But using that method for (0,0,0,0,2^{10000},0,0,0,0) will work very badly.

But, since $\frac{1}{\hat{n}_u} = max\{\frac{1}{n_u}, \frac{\epsilon}{2}\}$, we get that $\forall u \in U. \frac{1}{\hat{n}_u} \in \{\frac{1}{n_u}, \frac{\epsilon}{2}\}$.

Since $\hat{n}_u \geq 1$ (it is the number of nodes we visit at algorithm 2 - starting at one), we know that $\frac{1}{\hat{n}_u} \leq 1$, and also by definition $\hat{n}_u \leq \frac{2}{\epsilon}$, and therefore $\frac{1}{\hat{n}_u} \geq \frac{\epsilon}{2}$

We finally get that: $\frac{\epsilon}{2} \leq \frac{1}{\hat{n}_u} \leq 1$

Theorem 1:

$$Pr[|\widetilde{C} - \widehat{C}| \leq \frac{\epsilon}{2} \cdot \widehat{C}] \geq \frac{3}{4}$$

Proof: We will use the Chernoff bound:

a little reminder: in general, for $x_1, ..., x_r$ iid $x_i \in [0, 1]$ (actually we will even have: $x_i \in [\frac{\epsilon}{2}, 1]$)

if we consider $S = \Sigma x_i$, $p = E[x_i] = \frac{E[S]}{r}$, when using Chernoff multiplicative bound, we get:

$Pr[|\frac{S}{r} - p| \geq \delta p] \leq Pr[\frac{S}{r} \geq (1 + \delta) \cdot p] + Pr[\frac{S}{r} \leq (1 - \delta) \cdot p] \leq e^{-\frac{\delta^2 \cdot \mu}{3}} + e^{-\frac{\delta^2 \cdot \mu}{2}} \leq 2 \cdot e^{-\frac{\delta^2 \cdot \mu}{3}}$

Where the first inequality comes from union bound. The second from the two multiplicative Chernoff bounds, assuming $0 < \delta < 1$, and the third from adding them, taking into account that $\frac{1}{3} \leq \frac{1}{2} \Rightarrow e^{\frac{1}{3}} \leq e^{\frac{1}{2}} \Rightarrow e^{-\frac{1}{3}} \geq e^{-\frac{1}{2}} \Rightarrow e^{-\frac{\delta^2 \cdot \mu}{3}} \geq e^{-\frac{\delta^2 \cdot \mu}{2}}$

(An important remark: as long as each sample is chosen uniformly over n nodes, it's o.k if our values (that depend on r) does not seem independent (might as well we have a graph made of cliques of the same size, and all n_u are equivalent!) - In our case: as long as each U_i is chosen uniformly and all $\{U_i\}_{1 \leq i \leq r}$ are iid.)

So, when using this bound in our case we have: $p = E_{u \in U}[\frac{1}{\hat{n}_u}]$, $S = \Sigma_{i=1}^{r}(\frac{1}{\hat{n}_{u_i}})$, $\delta = \frac{\epsilon}{2}$

We also notice that: $E_{u \in U}\left[\frac{1}{\hat{n}_u}\right] = E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]$, since all v's in U are chosen uniformly and independently.

So, we finally get:

$$Pr\left[\left|\frac{1}{r}\Sigma_{i=1}^{r}\left(\frac{1}{\hat{n}_{u_i}}\right) - E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right| \geq \frac{\epsilon}{2} \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right] \leq 2 \cdot e^{-\frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3}}$$

We now want to find such r, so that the above probability would be bounded by $\frac{1}{4}$. So we follow this inequality for r:

$$2 \cdot e^{-\frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3}} \leq \frac{1}{4}$$

$$\Rightarrow e^{-\frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3}} \leq \frac{1}{8}$$

$$\Rightarrow \log(e^{-\frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3}}) \leq \log(\frac{1}{8})$$

$$\Rightarrow -\frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3} \leq -\log(8)$$

$$\Rightarrow \frac{(\frac{\epsilon}{2})^2 \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3} \geq \log(8)$$

$$\Rightarrow \frac{\epsilon^2}{4} \cdot \left(r \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right) \geq 3 \cdot \log(8)$$

$$\Rightarrow r \geq \frac{12 \cdot \log(8)}{\epsilon^2} \cdot \frac{1}{E\left[\frac{1}{\hat{n}_{u_i}}\right]}$$

Now, we notice that $E\left[\frac{1}{\hat{n}_{u_i}}\right] \geq \frac{\epsilon}{2}$, and therefore $\frac{1}{E\left[\frac{1}{\hat{n}_{u_i}}\right]} \leq \frac{2}{\epsilon}$

So it's enough that we take r s.t:

$$r \geq \frac{12 \cdot \log(8)}{\epsilon^2} \cdot \frac{2}{\epsilon} \geq \frac{12 \cdot \log(8)}{\epsilon^2} \cdot \frac{1}{E\left[\frac{1}{\hat{n}_{u_i}}\right]}$$

Therefore we get it's enough to take:

$$r_0 := \frac{24 \cdot \log(8)}{\epsilon^3}$$

We also know that $\frac{1}{r}\Sigma_{i=1}^{r}\left(\frac{1}{\hat{n}_{u_i}}\right) = \frac{\widetilde{C}}{n}$, and $E_{u \in V}\left[\frac{1}{\hat{n}_u}\right] = \frac{1}{n} \cdot \Sigma \frac{1}{\hat{n_u}} = \frac{\widehat{C}}{n}$

So, finally, taking such r_0 as we defined, we get:

$$Pr\left[|\widetilde{C} - \hat{C}| \geq \frac{\epsilon}{2} \cdot \hat{C}\right] = Pr\left[|\frac{\widetilde{C}}{n} - \frac{\hat{c}}{n}| \geq \frac{\epsilon}{2} \cdot \frac{\hat{c}}{n}\right] \leq 2 \cdot e^{-\frac{(\frac{\epsilon}{2})^2 \cdot \left(r_0 \cdot E_{u \in V}\left[\frac{1}{\hat{n}_u}\right]\right)}{3}} \leq \frac{1}{4}$$

Meaning, we have that:

$$Pr\left[|\widetilde{C} - \hat{C}| \leq \frac{\epsilon}{2} \cdot \hat{C}\right] \geq \frac{3}{4}$$

And thus, we proved Theorem 1!

So, with probability $\geq \frac{3}{4}$ we also have this inequality holding:

$$\left|\widetilde{C} - \widehat{C}\right| \leq \frac{\epsilon}{2} \cdot \widehat{C} \leq \frac{\epsilon n}{2}$$

$$\Rightarrow \left|\widetilde{C} - C\right| \leq \left|\widetilde{C} - \widehat{C}\right| + \left|\widehat{C} - C\right| \leq \frac{\epsilon n}{2} + \frac{\epsilon n}{2} \leq \epsilon n$$

when the second inequality in the first row comes from the fact that $\hat{C} \leq n$, and the second row's first inequality comes from the triangle inequality.