CS787: Advanced Algorithms **Scribe:** Group 4

Title: hw5

0.1 Heavy Hitters

The algorithm:
let $m = log(1/\eta); k = 2/\epsilon$;
let $Count[1...m][1...k] = 0$;
Select m independent hash functions from a 2-universal family: $h_1, h_2, ..., h_m : [n] \to [k]$
for $i : 1 \to m$:
for $j : 1 \to k$:
$Count[i][h_i(j)] = Count[i][h_i(j)] + 1/\theta$;
end
end
Return $f_q = max_{1 \le i \le m} Count[i][h(q)]$, on query q.
$Count[][]$ spend $log(1/\eta) \cdot log(2/\epsilon)$ space, while m hash functions spend $log(1/\eta)$ space. Thus the space complexity of this algorithm is $O(1/\epsilon \cdot log(1/\eta))$.

0.2 Counting In Small Space

Let X(m) denote random counter after m-th arrival, initialize X(0)=0; increment with probability $p_X = 2^{-X}$

(A)

$$\begin{aligned}
E[Y] = E[2^{X(m)}] &= \sum_{j=1}^{m-1} Pr[X(m-1) = j] * E[2^{X(m)} | \ X(m-1) = j] \\
&= \sum_{j=1}^{m-1} Pr[X(m-1) = j] * (p_j * 2^{j+1} + (1 - p_j) * 2^j) \\
&= \sum_{j=1}^{m-1} Pr[X(m-1) = j] * (2^j + 1) \\
&= E[2^{X(m-1)}] + 1 \\
&= E[2^{X(0)}] + m \\
&= 1 + m
\end{aligned}$$

(B)

$$
\begin{aligned}
E[2^{2X(m)}] &= \sum_{j=1}^{m-1} Pr[X(m-1)=j] * E[2^{2X(m)} |\ X(m-1)=j] \\
&= \sum_{j=1}^{m-1} Pr[X(m-1)=j] * (p_j * 2^{2j+2} + (1-p_j) * 2^{2j}) \\
&= \sum_{j=1}^{m-1} Pr[X(m-1)=j] * (2^{j+2} + 2^{2j} - 2^{j}) \\
&= \sum_{j=1}^{m-1} Pr[X(m-1)=j] * (3 * 2^{j} + 2^{2j}) \\
&= 3 * E[2^{X(m-1)}] + E[2^{2X(m-1)}] \\
&= 3m + E[2^{2X(m-1)}] \\
&= 3\sum_{l=1}^{m} l + E[2^{2X(0)}] \\
&= \frac{3m(m+1)}{2} + 1 \\
Var[Y] = Var[2^{X(m)}] &= E[2^{2X(m)}] - E[2^{X(m)}]^2 \\
&= (\frac{3m(m+1)}{2} + 1) - (m+1)^2 \\
&= \frac{m(m-1)}{2}
\end{aligned}
$$

(C) Let this process repeat k times. Denote Z = $\frac{1}{k}\sum_{i=1}^{k}(Y-1)$, we have $Var[Z] = \frac{1}{k}Var[Y]$. By Chebyshev's inequality:

$$
\begin{aligned}
Pr[|Z - E[Z]| \geq \epsilon E[Z]] &\leq \frac{Var[Z]}{(\epsilon E[Z])^2} \\
&= \frac{1}{m^2\epsilon^2 k}(\frac{1}{2}m^2 - \frac{1}{2}) \\
&\approx \frac{1}{2\epsilon^2 k}\ (Since\ m\ is\ much\ larger\ than\ 1)
\end{aligned}
$$

We want this probability to be δ, so k = $\frac{1}{2\epsilon^2\delta}$. Since storing X takes log(m) space, the total space would be $\frac{1}{2\epsilon^2\delta}log(m)$

0.3 Spanners

(A) For every $u, v \in H$ that are adjacent vertices in G: If (u,v) is added to H by algorithm, then $d_H(u,v) = d_G(u,v) = 1$. Else, we know that $d_G(u,v) = 1 \leq d_H(u,v) \leq t = t * d_G(u,v)$.

For every $u, v \in H$ that aren't adjacent in G: Let the shortest path from a to b in G/H be $P_G(a,b)/P_H(a,b)$, let its length be $|P_G(a,b)|/|P_H(a,b)|$. Assume $P_G(u,v) := u - x_1 - x_2 - ... - x_k - v$.
Since $(u, x_1), (x_i, x_{i+1}), (x_k, v)$ are adjacent in G, we will have:
$1 \leq |P_H(u, x_1)| \leq t$, $1 \leq |P_H(x_1, x_2)| \leq t$, ..., $1 \leq |P_H(x_k, v)| \leq t$

Thus we can get $k+1 = |P_G(u,v)| \leq |P_H(u,v)| \leq |P_H(u,x_1)| + |P_H(x_1,x_2)| + ... + |P_H(x_k,v)| = (k+1)t$, So, H is a t-spanner.

(B) If H doesn't have a circle, then girth of H is ∞. Else, we pick a random circle C from H. Let (u, v) be the last edge that was added to H by the algorithm and let its length be L. Before the iteration we decide whether or not to add (u,v), there is a path from u to v with length=L-1(the rest of the circle). So $d_H(u,v) \leq L - 1$. And, since (u,v) is added to H, then before this iteration, $t < d_H(u,v)$. So we have $t < L - 1 => t + 2 \leq L$ for any circle. Thus we can get $g \geq t + 2$.

(C)

0.4 Counting Min Cuts

The denotations are as below:
n: The vertex amount.
m: The edge amount.
k: The min cut, $|C^*|$
E_i: The event that at i iteration the ALG choose one of the edges from C. $(i \geq 0)$

(A) At the start of i iteration, there hence are n-i vertices, and there are at least $\frac{1}{2}k(n-i)$ edges. Hence,

$$P(E_i) \leq \frac{\alpha k}{\frac{1}{2}k(n-i)} = \frac{2\alpha}{n-i}$$

(B)

$$\begin{aligned}
&P(\neg E_0 \cap \neg E_1 \cap \neg E_2 \cap ... \cap \neg E_{end}) \\
&\quad = P(\neg E_0) * P(\neg E_1 | \neg E_0) * P(\neg E_2 | \neg E_0 \cap \neg E_1) * ... * P(\neg E_{end} | \neg E_0 \cap \neg E_1 \cap ... \cap \neg E_{end-1}) \\
&\quad \geq (1 - \frac{2\alpha}{n}) * (1 - \frac{2\alpha}{n-1}) * ... * (1 - \frac{2\alpha}{2\alpha + 1}) \\
&\quad = (\frac{n - 2\alpha}{n}) * (\frac{n - 1 - 2\alpha}{n-1}) * ... * (\frac{1}{2\alpha + 1}) \\
&\quad = \frac{2\alpha * (2\alpha - 1) * ... * 1}{n * (n-1) * ... * (n - 2\alpha + 1)} \\
&\quad \geq \frac{1}{n^{2\alpha}}
\end{aligned}$$

The above derivation shows that when the graph is decreased to $2\alpha + 1$ vertices, we should break the iteration and output the min cut among all $2^{2\alpha+1}$ cuts.

(C) From (b), we can get the upper bound of distinct cuts fewer than $\alpha|C^*|$ edges is $n^{2\alpha}$.

0.5 List Coloring

Let $e = (u, v) \in E$ be an edge in G.
And let ε_e be the case that e is monochromatic.
When $S(u) = S(v)$, the probability that e is monochromatic is the highest, so we can get $P[\varepsilon_e] \leq \frac{1}{k}$

Let Γ_e be the set of edges that intersect with e. Because u shares a color with at most k/10 other vertices, so there can at most be $\frac{k}{10} - 1$ other edges that e dependents on. So we can get $|\Gamma_e| \leq \frac{k}{10} - 1$

Because the edge can not be monochromatic if there are no shared colors, so all other edges not in Γ_e are independent of e. Because $e\frac{1}{k}(\frac{k}{10} - 1 + 1) = \frac{e}{10} < 1$, $P[\bigcup_{e \in E} \varepsilon_e] < 1$, thus we can apply the Lovasz Local Lemma to find that there always exists a proper list-coloring of G.