CMPSCI 690RA: Randomized Algorithms
Lecture 21: Graph Streaming and Sketching

Andrew McGregor
Motivating Problem

- **Problem:** There are $n$ machines and each has the row of an adjacency matrix of a graph with $n$ nodes. A single message is communicated from each machine to a central machine. How many bits do these messages need to be such that the central machine can determine whether the graph is connected?
- **Answer:** $O(\text{polylog } n)$ bits suffice such that the connectivity can be determined with high probability.
- **Corollary:** $O(n \text{ polylog } n)$ bits suffice to determine whether a graph defined by a stream of edge insertions/deletions is connected.
First Ingredient: Sketching for $\ell_0$ Sampling

Lemma

There exists random matrix $A \in \mathbb{R}^{O(\log^2 N) \times N}$ such that for any $x \in \mathbb{R}^N$, with probability at least $1 - 1/\text{poly}(n)$, we can learn $(i, x_i)$ for some $x_i \neq 0$ from $Ax$.

Useful properties:

- **Union Bound**: Suppose we have multiple vectors $x_1, x_2, \ldots, x_t$, then we can determine a non-zero element from everyone of them from

  $$Ax_1, Ax_2, \ldots, Ax_t$$

  with probability at least $1 - \delta t$.

- **Linearity**: Given $Ax$ and $Ay$, we can find a non-zero entry from $z = x + y$ since

  $$Az = A(x + y) = Ax + Ay$$
Consider the following (non-streaming) algorithm for connectivity:

- For each node, select an incident edge.
- For each connected component, select an incident edge.
- Repeat above line until process terminates.

Analysis:

- There are at most \( \log n \) rounds since in each round, the size of every connected component either stops growing or doubles size.
- The set of all edges selected includes a spanning forest of the graph.
Third Ingredient: Signed Vertex-Edge Vectors

With each vertex $i$ of the graph, associate a length $\binom{n}{2}$ vector that is indexed by pairs on nodes. The only non-zero entries correspond to incident edges $\{i, j\} \in E$ and this entry is $1$ if $j > i$ and $-1$ if $j < i$. E.g.,

\[
\begin{align*}
x_1 &= (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
x_2 &= (-1, 0, 0, 0, 1, 0, 0, 0, 0, 0) \\
x_3 &= (0, -1, 0, 0, -1, 0, 0, 1, 0, 0) \\
x_4 &= (0, 0, 0, 0, 0, 0, 0, -1, 0, 1) \\
x_5 &= (0, 0, 0, 0, 0, 0, 0, 0, 0, -1)
\end{align*}
\]

corresponds to a graph with edges $\{1, 2\}, \{1, 3\}, \{2, 3\}, \{3, 4\}$, and $\{4, 5\}$.

**Lemma**

Non-zero entries of $\sum_{i \in S} a_i$ correspond to edges between $S$ and $V \setminus S$.

**Proof.**

$\{j, k\}$th entry of $\sum_{i \in S} a_i$ equals $0$ iff $j, k \in S$ or $j, k \notin S$.  

\[\square\]
The Final Recipe

- **What players send:** Player with node $i$ sends $A_1x_i, A_2x_i, \ldots, A_{\log n}x_i$ where $A_1, A_2, \ldots$ are independent random matrices for $\ell_0$ sampling.

- **Central player emulates Boruvka’s algorithm:**
  - Can identify an incident edge from each node $i$ using $A_1x_i$ since can find a non-zero entry of $x_i$ and such entries of $x_i$ are incident edges.
  - In round $t$, suppose we need to find an incident edge from a connected component $S$. Then, we can such an edge since

$$
\sum_{i \in S} A_t x_i = A_t \sum_{i \in S} x_i
$$

and we can therefore identify of non-zero elements of $\sum_{i \in S} x_i$ which gives a suitable edge.
Basic idea for how $\ell_0$ sketching works

- Let $S_0, S_1, \ldots, S_{\log N}$ be random subsets of $[N]$ where each element is in $S_i$ with probability $1/2^i$.
- To sketch the vector $x$, for each $S \in \{S_0, S_1, \ldots, S_{\log N}\}$ compute:
  
  $$a = \sum_{j \in S} jx_j \quad b = \sum_{j \in S} x_j \quad c = \sum_{j \in S} x_j r^j \mod p$$

  where $r$ is a random value in range $1, \ldots, p - 1$ and $p = \text{poly}(N)$.
- We say $S$ passes the test if $a/b \in [N]$ and $c = br^{a/b} \mod p$.
  - If all $S$ do not pass the test, output “fail”
  - Otherwise, pick a passing $S$. Claim that $(a/b)$th entry of $x$ is $b > 0$
Lemma
Let $A = \{i \in N : x_i \neq 0\}$ be the positions of non-zero entries.

- If $|A \cap S| = 1$, then $S$ passes the test and $x_{a/b} = b$.
- If $|A \cap S| \neq 1$, then $S$ doesn't pass the test with high probability.

Proof.

- If $A \cap S = \{j\}$ then $a = jx_j$, $b = x_j$, and $c = bz^j \mod p$.
- If $|A \cap S| > 1$ then

$$f(z) = \sum_{j \in S} x_j z^j - bz^{a/b} \mod p$$

is a non-zero polynomial of degree at most $N$. Hence, it evaluates to 0 at a random $r$ with probability at most $N/(p-1) < 1/poly(N)$.

\[ \Box \]
Analysis: Part 2

**Lemma**

\[ \mathbb{P} [ |A \cap S| = 1 ] \geq 1/8 \text{ for some } S. \]

**Proof.**

Pick \( i \) such that \( 2^{i-2} \leq |A| < 2^{i-1} \). Then,

\[
\mathbb{P} [ |A \cap S_i| = 1 ] = \sum_{j \in A} \mathbb{P} [ j \in S_i, k \notin S_i \text{ for all } k \in A \setminus \{j\}] \\
= \sum_{j \in A} \frac{1}{2^i} \left( 1 - \frac{1}{2^i} \right)^{|A|-1} \\
= \frac{|A|}{2^i} \left( 1 - \frac{1}{2^i} \right)^{|A|-1} \\
> \frac{|A|}{2^i} \left( 1 - \frac{|A|}{2^i} \right) > 1/8
\]

Can boost the probability from \( 1/8 \) to \( 1 - 1/\text{poly}(n) \) by repeating the process \( O(\log n) \) times in parallel.
How to do it with hash functions: Part 1

Definition
We say a collection $\mathcal{H}$ of functions $D \rightarrow R$ is $k$-wise independent if for any set of $k$ distinct values $x_1, \ldots, x_k \in D$ and $k$ values $j_1, \ldots, j_k$ when we pick a function $h$ uniformly at random from $\mathcal{H}$,

$$\mathbb{P}[h(x_1) = j_1, h(x_2) = j_2, \ldots, h(x_k) = j_k] = 1/|R|^k$$

For example,

$$\mathcal{H} = \{h(x) = a_k x^k + a_{k-1} x^{k-1} + \ldots + a_0 \mod p : a_i \in \{0, 1, \ldots, p-1\} \text{ for all } i\}$$

is a family of $k$-wise hash functions from $[n]$ to $\{0, \ldots, p-1\}$ if $p$ a prime greater than $n$. Can store $h$ using $O(k \log p)$ bits.
How to do it with hash functions: Part 2

- To define $S_0, S_1, S_2, \ldots$, pick $h$ from a 2-wise independent family of hash functions.
- Let $S_i = \{x \in [N] : h(x) \text{ is divisible by } 2^i\}$ and so

$$\gamma_i = \mathbb{P}[j \in S_i] = \left(\left\lfloor \frac{(p - 1)}{2^i} \right\rfloor + 1 \right)/p \approx 1/2^i$$

- If $i$ satisfies that $2^{i-2} \leq |A| < 2^{i-1}m$ then,

$$\mathbb{P}[|A \cap S_i| = 1] = \sum_{j \in A} \mathbb{P}[j \in S_i, k \notin S_i \text{ for all } k \in A \setminus \{j\}]$$

$$= \sum_{j \in A} \gamma_i \mathbb{P}[k \notin S_i \text{ for all } k \in A \setminus \{j\}|j \in S_i]$$

$$\geq \sum_{j \in A} \gamma_i (1 - \sum_{k \in A \setminus \{j\}} \mathbb{P}[k \notin S_i|j \in S_i])$$

$$\geq \sum_{j \in A} \gamma_i (1 - \gamma_i) > 1/8$$
Assuming availability of random bits, each message can be computed in $O(\text{polylog } n)$ bits in the data stream model. Total of $O(n \text{polylog } n)$ bits.

When edge $\{i, j\}$ is inserted where $j > i$:

$$A_t x_i \leftarrow A_t x_i + A_t e_{i,j}$$
$$A_t x_j \leftarrow A_t x_j - A_t e_{i,j}$$

where $e_{i,j}$ is the length $\binom{n}{2}$ binary vector whose only non-zero entry is in the $\{i,j\}$th entry.

When edge $\{i, j\}$ is deleted where $j > i$:

$$A_t x_i \leftarrow A_t x_i - A_t e_{i,j}$$
$$A_t x_j \leftarrow A_t x_j + A_t e_{i,j}$$