LECTURE 7

Last time
• Communication complexity
• Other models of computation

Today
• Streaming

Project proposals due next Thursday
Sign up for project meetings, scribing, grading
**Data Stream Model** [Alon Matias Szegedy 96]

**Motivation:** internet traffic analysis

Model the stream as $m$ elements from $[n]$, e.g.,

$$\langle a_1, a_2, ... , a_m \rangle = 3, 5, 3, 7, 5, 4, ...$$

**Goal:** Compute a function of the stream, e.g., median, number of distinct elements, longest increasing sequence.

Based on Andrew McGregor’s slides: http://www.cs.umass.edu/~mcgregor/slides/10-jhu1.pdf
A stream contains \( n - 1 \) distinct elements from \([n]\) in arbitrary order.

**Problem:** Find the missing element, using \( O(\log n) \) space.
Sampling from a Stream of Unknown Length

Warm-up: Find a uniform sample $s$ from a stream $\langle a_1, a_2, \ldots, a_m \rangle$ of known length $m$. 
**Sampling from a Stream of Unknown Length**

**Problem:** Find a uniform sample $s$ from a stream $\langle a_1, a_2, ..., a_m \rangle$ of unknown length $m$

**Algorithm (Reservoir Sampling)**

1. Initially, $s \leftarrow a_1$
2. On seeing the $t^{th}$ element, $s \leftarrow a_t$ with probability $1/t$

**Analysis:**

What is the probability that $s = a_i$ at some time $t \geq i$?

$$\Pr[s = a_i] = \frac{1}{i} \cdot \left(1 - \frac{1}{i+1}\right) \cdot \ldots \cdot \left(1 - \frac{1}{t}\right)$$

$$= \frac{1}{i} \cdot \frac{i}{i+1} \cdot \ldots \cdot \frac{t-1}{t} = \frac{1}{t}$$

**Space:** $O(k \ (\log n + \log m))$ bits to get $k$ samples.
Counting Distinct Elements

Input: a stream $\langle a_1, a_2, \ldots, a_m \rangle \in [n]^m$

Warm-up: Output the number of distinct elements in the stream.

Exact solutions:
- Store $n$ bits, indicating whether each domain element has appeared.
- Store the stream: $O(m \log n)$ bits.

Known lower bounds:
- Every deterministic algorithm requires $\Omega(m)$ bits (even for a constant-factor approximation).
- Every exact algorithm (even randomized) requires $\Omega(n)$ bits.

Need to use both randomization and approximation to get $\text{polylog}(m, n)$ space.
Counting Distinct Elements

Input: a stream \( \langle a_1, a_2, \ldots, a_m \rangle \in [n]^m \)

Goal: Estimate the number of distinct elements in the stream up to a multiplicative factor \((1 + \varepsilon)\) with probability \(\geq 2/3\)

- Studied by [Flajolet Martin 83, Alon Matias Szegedy 96,...]

- **Today**: \(O(\varepsilon^{-2} \log n)\) space algorithm
  [Bar–Yossef Jayram Kumar Sivakumar Trevisan 02]

- **Optimal**: \(O(\varepsilon^{-2} + \log n)\) space algorithm [Kane Nelson Woodruff 10]
Counting Distinct Elements

Input: a stream \( \langle a_1, a_2, \ldots, a_m \rangle \in [n]^m \)

Goal: Estimate the number of distinct elements in the stream up to a multiplicative factor \((1 + \varepsilon)\) with probability \(\geq 2/3\)

Algorithm

1. Apply a random hash function \( h : [n] \rightarrow [n] \) to each element
2. Compute \( X \), the \( t \)-th smallest value of the hash seen where \( t = \frac{10}{\varepsilon^2} \)
3. Return \( \tilde{r} = \frac{t \cdot n}{X} \) as estimate for \( r \), the number of distinct elements.

Analysis:

- Algorithm uses \( O(\varepsilon^{-2} \log n) \) bits of space (not accounting for storing \( h \))
- We'll show: estimate \( \tilde{r} \) has good accuracy with reasonable probability.

Claim. \( \Pr[|\tilde{r} - r| \leq \varepsilon r] \geq 2/3 \)
Counting Distinct Elements: Analysis

**Claim.** \( \Pr[|\tilde{r} - r| \leq \varepsilon r] \geq 2/3 \)

**Proof:** Suppose the distinct elements are \( e_1, \ldots, e_r \)

- **Overestimation:**
  \[
  \Pr[\tilde{r} \geq (1 + \varepsilon)r] = \Pr \left[ \frac{t \cdot n}{X} \geq (1 + \varepsilon)r \right] = \Pr \left[ X \leq \frac{t \cdot n}{r(1 + \varepsilon)} \right]
  \]

- Let \( Y_i = \mathbb{1}[h(e_i) \leq \frac{t \cdot n}{r(1 + \varepsilon)}] \) and \( Y = \sum_{i=1}^{r} Y_i \)
  \[
  E[Y] = r \cdot E[Y_1] = r \cdot \frac{t}{r(1 + \varepsilon)} = \frac{t}{1 + \varepsilon}
  \]
  \[
  \text{Var}[Y] = \text{Var} \left[ \sum_{i=1}^{r} Y_i \right] = \sum_{i=1}^{r} \text{Var}[Y_i] \leq \sum_{i=1}^{r} E[Y_i^2] = \sum_{i=1}^{r} E[Y_i] = E[Y]
  \]

\( X: t\)-th smallest hashed value
\[
\tilde{r} = t \cdot n/X
\]

\[
E[Y] = \frac{t}{1 + \varepsilon}
\]
\[
\text{Var}[Y] \leq E[Y]
\]
Counting Distinct Elements: Analysis

Claim. \( \Pr[|\tilde{r} - r| \leq \varepsilon r] \geq 2/3 \)

Proof: Suppose the distinct elements are \( e_1, \ldots, e_r \)

- Overestimation:
  \[
  \Pr[\tilde{r} \geq (1 + \varepsilon)r] = \Pr \left[ \frac{t \cdot n}{X} \geq (1 + \varepsilon)r \right] = \Pr \left[ X \leq \frac{t \cdot n}{r(1 + \varepsilon)} \right]
  \]

- Let \( Y_i = \mathbb{1} \left[ h(e_i) \leq \frac{t \cdot n}{r(1+\varepsilon)} \right] \) and \( Y = \sum_{i=1}^{r} Y_i \)

  \[
  \Pr \left[ X \leq \frac{t \cdot n}{r(1 + \varepsilon)} \right] = \Pr[Y \geq t] = \Pr[Y \geq (1 + \varepsilon)E[Y]]
  \]

- By the Chebyshev's inequality, for \( \varepsilon \leq 2/3 \),

  \[
  \Pr[Y \geq (1 + \varepsilon)E[Y]] \leq \frac{\text{Var}[Y]}{(\varepsilon \cdot E[Y])^2} \leq \frac{1}{\varepsilon^2 E[Y]} = \frac{1 + \varepsilon}{\varepsilon^2 t} = \frac{1 + \varepsilon}{10} \leq \frac{1}{6}
  \]

- Underestimation: A similar analysis shows \( \Pr[\tilde{r} \leq (1 - \varepsilon)r] \leq \frac{1}{6} \)

\( X \): \( t \)-th smallest hashed value

\( \tilde{r} = t \cdot n/X \)

\( t = 10 / \varepsilon^2 \)

\( E[Y] = \frac{t}{1 + \varepsilon} \)

\( \text{Var}[Y] \leq E[Y] \)
Removing the Random Hashing Assumption

Idea: Use limited independence

- A family $\mathcal{H} = \{h: [a] \to [b]\}$ of hash functions is $k$-wise independent if for all distinct $x_1, \ldots, x_k \in [a]$ and all $y_1, \ldots, y_k \in [b]$,

$$\Pr_{h \in \mathcal{H}} [h(x_1) = y_1, \ldots, h(x_k) = y_k] = \frac{1}{b^k}$$

Note: a uniformly random family is $k$-wise independent for all $k$

- Observations: For $x_1, \ldots, x_k$ as above,
  1. $h(x_1)$ is uniform over $[b]$
  2. $h(x_1), \ldots, h(x_k)$ are mutually independent.
Construction of $k$-wise Independent Family

Idea: Use limited independence

- A family $\mathcal{H} = \{ h: [a] \rightarrow [b] \}$ of hash functions is $k$-wise independent if for all distinct $x_1, ..., x_k \in [a]$ and all $y_1, ..., y_k \in [b]$,
  $$\Pr_{h \in \mathcal{H}}[h(x_1) = y_1, ..., h(x_k) = y_k] = \frac{1}{b^k}$$

**Construction of $k$-wise Independent Family of Hash Functions**

1. Let $p$ be a prime.
2. Consider the set of polynomials of degree $k - 1$ over $\mathbb{F}_p$
   $$\mathcal{H} = \{ h: \{0, ..., p - 1\} \rightarrow \{0, ..., p - 1\} \mid h(x) = c_{k-1}x^{k-1} + \cdots + c_1x + c_0, \text{with } c_0, ..., c_{k-1} \in \mathbb{F}_p \}$$
3. To sample $h \in \mathcal{H}$, sample $c_0, ..., c_{k-1} \in \mathbb{F}_p$ u.i.r.

- Space to store $h$ is $O(k \log p)$
- For arbitrary $a, b$, need $O(k \cdot (\log a + \log b))$ space.

Based on Sepehr Assadi’s lecture notes for CS 514 (Lecture 7, 03/20/20) at Rutgers
Counting Distinct Elements: Final Algorithm

Input: a stream \( \langle a_1, a_2, \ldots, a_m \rangle \in [n]^m \)

Goal: Estimate the number of distinct elements in the stream up to a multiplicative factor \( (1 + \varepsilon) \) with probability \( \geq 2/3 \)

### Algorithm

1. Sample a hash function \( h : [n] \rightarrow [n] \) from a 2-wise independent family and apply \( h \) to each element
2. Compute \( X \), the \( t \)-th smallest value of the hash seen where \( t = 10 / \varepsilon^2 \)
3. Return \( \tilde{r} = t \cdot n/X \) as estimate for \( r \), the number of distinct elements.

### Analysis:

- Algorithm uses \( O(\varepsilon^{-2} \log n) \) bits of space
- Our correctness analysis applies.
Frequency Moments Estimation

Input: a stream \( \langle a_1, a_2, \ldots, a_m \rangle \in [n]^m \)

- The frequency vector of the stream is \( f = (f_1, \ldots, f_m) \), where \( f_i \) is the number of times \( i \) appears in the stream

- The \( p \)-th frequency moment is \( F_p = \| f \|_p^p = \sum_{i=1}^{n} f_i^p \)
  
  \( F_0 \) is the number of nonzero entries of \( f \) (# of distinct elements)
  \( F_1 = m \) (# of elements in the stream)
  \( F_2 = \| f \|_2^2 \) is a measure of non-uniformity
  used e.g. for anomaly detection in network analysis
  \( F_\infty = \max_i f_i \) is the most frequent element

Goal: Estimate \( F_p \) up to a multiplicative factor \((1 + \varepsilon)\) with probability \( \geq 2/3 \)
Summary

Streaming Model
• Reservoir sampling
• Distinct Elements (approximating $F_0$)
• $k$-wise independent hashing