1 Review

Morris algorithm

- Initially: $X = 0$
- At increment: $X := X + 1$ with probability $1/2^X$
- Estimator: $E = 2^X - 1$

Morris+

- Run $k$ independent instances of Morris: $X_1, \ldots, X_k$
- Estimator: $E = \text{average estimator} = \frac{1}{k} \sum_{i=1}^{k} (2^{X_i} - 1)$

Claim 3. $\mathbb{E}[E] = n$

Claim 4. $\text{Var}[E] = \frac{3/2 \cdot n^2}{k}$ for $n \geq 10$

2 Improving concentration

From a Chebyshev bound, we obtain

$$\Pr[E \notin (n \pm \lambda)] \leq \frac{\text{Var}[E]}{\lambda^2} = \frac{3/2 \cdot n^2/k}{\lambda^2},$$

so we can set $\lambda = \epsilon n \Rightarrow k = 3/2 \cdot 10 \cdot 1/\epsilon^2$ to obtain 0.1 failure probability.

What if in general we want to get a success probability of $\geq 1 - \delta$, for some (small) parameter $\delta$? We need (1) to be $\leq \delta$, so we should set

$$k = 3/2 \cdot 10 \cdot 1/\epsilon^2 \cdot 1/\delta = \Theta(1/\epsilon^2 \cdot 1/\delta).$$

Can we get a better dependence on $1/\delta$? Yes, using the median trick.
**Median trick**  Goal: amplify the probability to be in the correct range, using the original algorithm as a black box.

<table>
<thead>
<tr>
<th>Algorithm $A \rightarrow \text{out}_A \in \mathbb{R}$</th>
<th>median trick</th>
<th>Algorithm $A^* \rightarrow \text{out}_{A^*} \in \mathbb{R}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Pr[\text{out}_A \in \text{correct range}] \geq 0.9$</td>
<td></td>
<td>$\Pr[\text{out}_{A^*} \in \text{correct range}] \geq 1 - \delta$</td>
</tr>
</tbody>
</table>

How it works:

- Run $k$ independent copies of $A$: $A_1, \ldots, A_k$
- Output $\text{out}_{A^*} = \text{median value of } A_1, \ldots, A_k$

We show that in this case, $k = O(\log 1/\delta)$ is enough.

**Chernoff/Hoeffding bound**  Let $X_1, \ldots, X_k$ be independent random variables $\in [0, 1]$, and let $\mu = \mathbb{E}\left[\sum_{i=1}^{k} X_i\right]$. For any $\epsilon \in [0, 1/2]$, we have

$$\Pr\left[\left|\sum_{i=1}^{k} X_i - \mu\right| > \epsilon \mu\right] \leq 2e^{-\epsilon^2 \mu/3}.$$

*Proof of median trick.* Let $X_i = \chi[A_i \text{ is correct}].$\(^1\) Clearly, $\mathbb{E}[X_i] = \Pr[X_i = 1] \geq 0.9$, so $\mu \geq 0.9 \cdot k$.

When is $A^*$ correct? Well, at any rate it is correct whenever $> 50\%$ of the $A$’s are correct, that is, when $\sum_{i=1}^{k} X_i > 0.5 \cdot k$.

Now, using a Chernoff bound with $\epsilon = 0.4$, we obtain

$$\Pr\left[\left|\sum_{i=1}^{k} X_i - 0.9 \cdot k\right| > 0.4 \cdot 0.9 \cdot k\right] \leq 2e^{-0.4^2 \cdot 0.9 \cdot k/3}.$$  

If the condition in $\Pr[]$ doesn’t hold, then we have $\sum_{i=1}^{k} X_i \geq 0.9 \cdot k - 0.4 \cdot 0.9 \cdot k > 0.5 \cdot k$, as desired. So we just have to make sure the RHS is $\leq \delta$, which we can achieve by setting

$$k = \frac{3}{0.4^2 \cdot 0.9 \ln(2/\delta)} = \Theta(\log 1/\delta).$$

This means that if we apply the median trick to Morris+ (which uses $O(1/\epsilon^2)$ instances to get a $(1 + \epsilon)$-approximation with probability $0.9$), then $O(1/\epsilon^2 \log 1/\delta)$ total instances are enough.

---

\(^1\)By $\chi[]$ we will denote the *characteristic* of a condition: the random variable that is 1 if the condition is true, and 0 if the condition is false.
3 Counting distinct elements

Consider a stream $X_1, \ldots, X_m \in [n]$ of IPs going through a router (think of $m$ and $n$ as very large and comparable in size).

Problem. Count the number of distinct $X_i$'s = $|\{X_1, \ldots, X_m\}|$ using little space.

Basic solutions

- Bit array of length $n$: for each value in $[n]$, has it been seen yet?
- Set structure, stored with $O(m \log n)$ bits.

Can we do better?

Flajolet-Martin [Flajolet-Martin'85]
Uses a hash function oracle $h : [n] \rightarrow [0, 1]$, where each $h(i)$ is an independently chosen random real number in $[0, 1]$.

- Initially: $Z = 1$
- On seeing $X = i$: $Z := \min(Z, h(i))$
- Estimator: $E = \frac{1}{Z} - 1$

Example. For stream 1, 3, 1, 7 and values of $h$ below, the algorithm will choose $Z = h(3)$.

Analysis of Flajolet-Martin

Let $d$ be the number of distinct elements in the stream.

Claim 1. $\mathbb{E}[Z] = \frac{1}{d+1}$

Observation. Repeats don’t affect $Z$, so we can consider that the stream is actually composed of $d$ distinct elements. Therefore $Z$ is the minimum of $d$ random variables distributed independently and uniformly in $[0, 1]$.

Proof of claim 1. Pick a fresh variable $A \in [0, 1]$ at random. Consider the probability $\Pr[A < Z]$.

- On the one hand, we clearly have $\Pr[A < Z] = \mathbb{E}[Z]$.
- On the other hand, note that $A, X_1, \ldots, X_d$ are $d+1$ iid variables, and $A < Z$ is true iff $A$ is the smallest of them all. Since the probability of a tie is 0, by symmetry we have $\Pr[A < Z] = \frac{1}{d+1}$. 

The claim follows from combining both equalities.

**Claim 2.** \( \text{Var}[Z] \leq \frac{2}{d^2} \)

**Proof.** Skipped.

Even with those guarantees, there is a big issue: \( \mathbb{E}[1/Z] \neq 1/\mathbb{E}[Z] \) in general, and they can wildly differ. So how do we get a \((1 + \epsilon)\)-approximation anyway? Two options.

**Option 1** (Flajolet-Martin+).
- Run \( k \) iid FM instances \( Z_1, \ldots, Z_k \)
- Estimator: \( \frac{1}{Z} - 1 \), where \( Z = \frac{1}{k} \sum_{i=1}^{k} Z_i \)

We claim that \( k = O(1/\epsilon^2) \) is enough.

**Proof.** Skipped.

**Option 2** (Bottom-\( k \) algorithm). [BJKS'02]
- Assumes that \( d \) is sufficiently large (\( > k \)). Uses only one hash function \( h \) (instead of \( k \) for option 1).
  - Initially: \( Z_1 = \cdots = Z_k = 1 \)
  - Maintain \( Z_1 < Z_2 < \cdots < Z_k \): the \( k \) smallest hash function values seen so far
  - Estimator: \( \hat{d} = \frac{k}{Z_k} \)

**Analysis of bottom-\( k \) algorithm**

**Lemma 3.** \( \Pr[\hat{d} > d(1 + \epsilon)] \leq 0.05 \) and \( \Pr[\hat{d} < d/(1 + \epsilon)] \leq 0.05 \).

**Proof of first part.** Without loss of generality, we can assume the stream is just \( 1, 2, \ldots, d \). Let \( X_i := \chi[h(i) < \frac{k}{(1+\epsilon)d}] \). We assume that \( \frac{k}{(1+\epsilon)d} \leq 1 \).

**Observation.** \( \hat{d} > d(1 + \epsilon) \Leftrightarrow Z_k < \frac{k}{(1+\epsilon)d} \Leftrightarrow \sum_{i=1}^{d} X_i \geq k \).

- \( \mathbb{E}\left[ \sum_{i=1}^{d} X_i \right] = d \cdot \mathbb{E}[X_i] = d \frac{k}{(1+\epsilon)d} = \frac{k}{1+\epsilon} \)
- \( \text{Var}\left[ \sum_{i=1}^{d} X_i \right] = d \cdot \text{Var}[X_i] \leq d \cdot \mathbb{E}[X_i^2] = d \frac{k}{(1+\epsilon)d} = \frac{k}{1+\epsilon} \leq k \)

By a Chebyshev bound,

\[
\Pr\left[ \sum_{i=1}^{d} X_i - \frac{k}{1+\epsilon} > \sqrt{20k} \right] \leq 0.05,
\]

where the inequality inside \( \Pr[] \) is equivalent to

\[
\sum_{i=1}^{d} X_i > \frac{k}{1+\epsilon} + \sqrt{20k}.
\]
The RHS is at most
\[ k(1 - \epsilon + \epsilon^2) + \sqrt{20k} \leq k \]
as long as \( \epsilon < 1/2 \) and \( k > \frac{25}{\epsilon - \epsilon^2} = \Theta(1/\epsilon^2) \), so \( \Pr\left[ \sum_{i=1}^{d} X_i \geq k \right] \leq 0.05. \)

Therefore, we have the following:

**Theorem 4.** For any \( \epsilon < \frac{1}{2} \), for \( d = \Omega(1/\epsilon^2) \), the bottom-\( k \) algorithm has a space complexity of \( O(1/\epsilon^2) \) counters.

**Relaxing requirements for the hash function**

We make two observations.

- We only care about order, (the absence of) collisions, and approximate values. So it’s fine to use something like
  \[
  h : [n] \to \left\{ 0, \frac{1}{M}, \frac{2}{M}, \ldots, \frac{M - 1}{M}, 1 \right\}
  \]
  instead of reals in \([0, 1]\). If \( M \gg n^3 \), then there are no collisions with probability \( \geq 1 - 1/n \). In this regime, the counters only take up \( O(\log M) = O(\log n) \) bits.

- 2-wise independence for the hash function is enough: no need for \( n \)-wise independence. Indeed, all we used was
  - computations like \( \mathbb{E}[X_i] = \Pr[h(i) < \cdots] = \cdots \);
  - the fact that terms like \( \mathbb{E}[X_i X_j] (i \neq j) \) disappear in \( \text{Var}\left[ \sum_{i=1}^{d} X_i \right] \).