CS60021: Scalable Data Mining

Streaming Algorithms

Sourangshu Bhattacharya

Count distinct

Streaming problem: distinct count

- Universe is U, number of distinct elements = m, stream size is n
  - Example: $U =$ all IP addresses

    10.1.21.10, 10.93.28,1,….,98.0.3.1,…..10.93.28.1…..

  - IPs can repeat
  - Want to estimate the number of distinct elements in the stream

Other applications

- Universe = set of all k-grams, stream is generated by document corpus
  - need number of distinct k-grams seen in corpus

- Universe = telephone call records, stream generated by tuples (caller, callee)
  - need number of phones that made > 0 calls

Solutions

- Seen n elements from stream with elements from U.

- Naïve solution: $O(n\ log|U|)$ space
  - Store all elements, sort and count distinct
  - Store a hashmap, insert if not present

- Bit array: O(|U|) space:
  - Bits initialized to 1 only if element seen in stream

- Can we do this in less space ?

Approximations

- (ϵ, δ) −approximations
  - Algorithm will use random hash functions
  - Will return an answer $\hat{n}$ such that

$$(1-\epsilon)n \leq \hat{n} \leq (1+\epsilon)n$$

  - This will happen with probability $1-\delta$ over the randomness of the algorithm

First effort

- Stream length: n, distinct elements: m
- Proposed algo: Given space s, sample s items from the stream
  - Find the number of distinct elements in this set: $\widehat{m}$
  - return $\mathrm{m} = \widehat{m} \times \frac{n}{s}$
- Not a constant factor approximation
  - $\underbrace{1, 1, 1, 1, \ldots, 1}_{n-m+1}, 2, 3, 4, \ldots, m\text{-}1$

Linear Counting

- Bit array B of size m, initialized to all zero
- Hash function $h: U \to [m]$
- When seeing item x , set $B[h(x)] = 1$

Linear Counting

- Bit array B of size m, initialized to all zero
- Hash function $h: U \to [m]$
- When seeing item x , set $B[h(x)] = 1$

- $z_m =$ fraction of zero entries
- Return estimate $-\mathrm{m} \log(\frac{z_m}{m})$

Linear Counting Analysis

- Pr[position remaining 0] = $\left(1 - \frac{1}{m}\right)^n \approx e^{-\frac{n}{m}}$
- Expected number of positions at zero: $\mathrm{E}[\mathrm{z_m}] = me^{-n/m}$

- Using tail inequalities we can show this is concentrated
- Typically useful only for $m = \Theta(n)$, often useful in practice

Flajolet Martin Sketch

Flajolet Martin Sketch

- Components
  - "random" hash function $h: U \rightarrow 2^{\ell}$ for some large ℓ
  - $h(x)$ is a ℓ −length bit string
  - initially assume it is completely random, can relax
- $zero(v)$ = position of rightmost 1 in bit representation of v

  $= \max\{\, i\,, 2^{i}\ divides\ v\,\}$

  - $zeros(10110) = 1, \quad zeros(110101000) = 3$

Flajolet Martin Sketch

<u>Initialize:</u>

– Choose a “random” hash function $h: U \to 2^{\ell}$

– $z \leftarrow 0$

<u>Process(x)</u>

– if $zeros\big(h(x)\big) > z,\ \ \mathrm{z} \leftarrow zeros(h(x))$

<u>Estimate:</u>

– return $2^{z+1/2}$

Example

	h(.)
	0110101
	1011010
	1000100
	1111010

Space usage

- We need $\ell \geq C \log(n)$ for some $C \geq 3$, say
  - by birthday paradox analysis, no collisions with high prob

- Sketch : z , needs to have only $O(\log \log n)$ bits

- Total space usage = $O(\log n + \log \log n)$

Intuition

- Assume hash values are uniformly distributed
- The probability that a uniform bit-string
  - is divisible by 2 is ½
  - is divisible by 4 is ¼
  - ….
  - is divisible by 2^k is $\frac{1}{2^k}$
- We don't expect any of them to be divisible by $2^{\log_2(n)+1}$

Formalizing intuition

- $S =$ set of elements that appeared in stream
- For any $r \in [\ell], j \in [n]$, $X_{rj} =$ indicator of $\mathrm{zeros}(h(j)) \geq r$
- $Y_r =$ number of $j \in U$ such that $\mathrm{zeros}(h(j)) \geq r$

$$Y_r = \sum_{j \in S} X_{rj}$$

- Let $\hat{z}$ be final value of z after the algorithm has seen all data

Proof of FM

- $Y_r > 0 \leftrightarrow \hat{z} \geq r$, equivalently, $Y_r = 0 \leftrightarrow \hat{z} < r$
- $E[Y_r] = \sum_{j \in S} E\left[X_{rj}\right]$ $\qquad X_{rj} = \begin{cases} 1 \;\; with\, prob \;\; \dfrac{1}{2^r} \\ 0 \qquad\qquad\quad else \end{cases}$
- $E[Y_r] = \frac{n}{2^r}$
- $var(Y_r) = \sum_{j \in S} var\left(X_{rj}\right) \;\; \leq \sum_{j \in S} E\left[X_{rj}^2\,\right]$

Proof of FM

- $var(Y_r) \leq \sum_{j \in S} E\left[X_{rj}^2\right] \leq \ n/2^r$

$$\Pr[Y_r > 0] = \Pr[Y_r \geq 1] \leq \frac{E[Y_r]}{1} = \frac{n}{2^r}$$

$$\Pr[Y_r = 0] \leq \Pr\left[\,|Y_r - E[Y_r]| \geq E[Y_r]\right] \leq \frac{var(Y_r)}{E[Y_r]^2} \leq \frac{2^r}{n}$$

Upper bound

Returned estimate $\hat{n} = 2^{\hat{z}+1/2}$

a = smallest integer with $2^{a+1/2} \geq 4n$

$$\Pr[\hat{n} \ \geq 4n\ ] = \Pr[\ \hat{z} \geq a\ ] = \Pr[Y_a > 0] \leq \frac{n}{2^a} \leq \frac{\sqrt{2}}{4}$$

Lower bound

Returned estimate $\hat{n} = 2^{\hat{z}+1/2}$

b = largest integer with $2^{b+1/2} \leq n/4$

$$\Pr\left[\hat{n} \leq \frac{n}{4}\right] = \Pr[\ \hat{z} \leq b\ ] = \Pr[Y_{b+1} = 0] \leq \frac{2^{b+1}}{n} \leq \frac{\sqrt{2}}{4}$$

Understanding the bound

- By union bound, with prob $1 - \frac{\sqrt{2}}{2}$,
$$\frac{n}{4} \leq \hat{n} \leq 4n$$

- Can get somewhat better constants
- Need only 2-wise independent hash functions, since we only used variances

Improving the probabilities

- To improve the probabilities, a common trick: median of estimates

- Create $\widehat{z_1}, \widehat{z_2}, \ldots., \widehat{z_k}$ in parallel
  – return median
- Expect at most $\frac{\sqrt{2}}{4}k$ of them to exceed $4n$
- But if median exceeds $4n$, then $\frac{k}{2}$ of them does exceed 4n
  → using this prob is $\exp(-\Omega(k))$

Improving the probabilities

- To improve the probabilities, a common trick: median of estimates

- Create $\widehat{z_1}, \widehat{z_2}, \ldots., \widehat{z_k}$ in parallel
  - return median

- Using Chernoff bound, can show that median will lie in $\left[\frac{n}{4}, 4n\right]$ with probability $1 - \exp(-\Omega(k))$.

- Given error prob δ, choose $k = O(\log\left(\frac{1}{\delta}\right))$

k-minimum value Sketch

k-MV sketch

- Developed in an effort to get better accuracy
  - Flajolet Martin only give multiplicative accuracy

- Additional capabilities for estimating cardinalities of union and intersection of streams
  - If S_1 and S_2 are two streams, can compute their union sketch from individual sketches of S_1 and S_2

[kMV sketch slides courtesy Cohen-Wang]

Intuition

- Suppose $h: U \to [0,1]$ is random hash function such that $h(x) \sim U[0,1]$ for all $x \in U$
- Maintain min-hash value y
  - initialize $y \leftarrow 1$
  - For each item x_i, $y \leftarrow \min(y, h(x_i))$
- Expectation of minimum is $E[\min_i h(x_i)] = \frac{1}{n+1}$

Why is expectation of min = $\frac{1}{n+1}$?

Intuition:

- You have sampled n points uniformly at random in interval [0,1]
- $n + 1$ intervals are formed.
- Expected length of each interval is $\frac{1}{n+1}$
- Value of E$[\min_i X_i]$ is the length of an interval.

Why is expectation of min = $\frac{1}{n+1}$?

Assuming a $X_i = U(0,1)$, we have:

$$P\left(\min_i X_i \le x\right) = 1 - P\left(\min_i X_i \ge x\right) = 1 - (1-x)^n$$

So, the density function is: $f(x) = n(1-x)^{n-1}$

Hence,

$$E[\min_i X_i] = \int_0^1 x f(x) dx = n \int_0^1 x(1-x)^{n-1} dx = \frac{1}{n+1}$$

k-minimum value sketch

Initialize:

- $y_1, \ldots, y_k \leftarrow 1, \ldots 1$
- Uniform random hash functions $h_1, \ldots, h_k$, $h_i: U \rightarrow [0,1]$

Process(x):

- For all $j \in [k]$, $y_j \leftarrow \min(y_j, h_j(x_i))$

Estimate:

- return median-of-means($\frac{1}{y_1}, \ldots, \frac{1}{y_k}$)

Example

	h1	h2	h3	h4
	.45	.19	.10	.92
	.35	.51	.71	.20
	.21	.07	.93	.18
	.14	.70	.50	.25

Median-of-means

- Given (ϵ, δ) , choose $k = \frac{c}{\epsilon^2}\log(\frac{1}{\delta})$

- Group $t_1, \ldots t_k$ into $\log(\frac{1}{\delta})$ groups of size $\frac{c}{\epsilon^2}$ each

- Find $\text{mean}(t_i)$ for each group: $Z_1, \ldots, Z_{\log(\frac{1}{\delta})}$

- Return $\hat{n} =$ median of $Z_1, \ldots Z_{\log(\frac{1}{\delta})}$

Complexity

- Total space required =
$O(k \log n) = O(\frac{1}{\epsilon^2} \log n \log(\frac{1}{\delta}))$
  – can be improved
  – don't need floating points, can use $h: U \to 2^{\ell}$ as before

- Update time per item = $O(k)$
  – However, can show that most items will not result in updates

Theoretical Guarantees

With probability $1-\delta$, returns $\hat{n}$ satisfies

$$(1\ -\epsilon)n \leq \hat{n} \leq (1+\epsilon)n$$

Proof: Apply Chebychef's inequality

$$P(|X_N - \mu_X| \geq \epsilon) \leq \frac{\sigma_X^2}{N\epsilon^2} \Rightarrow N \geq \frac{\sigma_X^2}{\epsilon^2}$$

followed by Chernoff bounding.

Merging

- For two stream S_1 and S_2 use same set of hash functions
  - Stream S_i has sketch $(y_1^i, \dots, y_k^i)$

- For each j $\in [k]$, find the combined sketch as:
  - $\mathrm{y_j} = \min(y_j^1, y_j^2)$

- Gives estimate of $|S_1 \cup S_2|$

References:

- Primary reference for this lecture
    - Lecture notes by Amit Chakrabarti: http://www.cs.dartmouth.edu/~ac/Teach/data-streams-lecnotes.pdf