CS787: Advanced Algorithms	**Scribe:** Zirui Tao
Lecture 20: Streaming Algorithm/ Heavy Hitter	**Date:** April 11, 2019

20.1 Setting and notation

1. Suppose n, m are two positive integer with $n \leq m$. Given universe U, where $|U| = n$ and a stream $i_1, i_2, \dots i_m$ with $i_j \in U, \forall j \in [m]$.
2. Algorithm can process each element i_j exactly once in sequence.
3. Only allowed to use very constrained space $O(polylog(m, n))$, in which "$polylog(\cdot)$" stands for poly logarithmic.

Goal: for a quantity Q, an (ϵ, δ) approximation is an estimator X such that

$$\mathbb{P}\left[x \in (1 \pm \epsilon)Q \mid X = x\right] \geq 1 - \delta$$

Before proceeding, further denote the following quantities:

- $f(x)$: For $x \in U$, denote $f(x) = |\{t : i_t = x\}|$. Thus, $f(x)$ measures the number of occurrence of distinct element $x \in U$ throughout the entire stream $\{i_j\}_{j=1}^m$.
- $F(x)$: Given a fixed stream $\{i_j\}_{j=1}^m$, $F(x)$ measures the frequency of the element $x \in U$ in this stream: $F(x) := \frac{f(x)}{\sum_{y \in U} f(y)}$.
- **majority**: An element $x \in U$ is a majority $\iff F(x) > \frac{1}{2}$.

20.2 Heavy hitter

Here, we start the analysis with proposing an algorithm called HEAVY HITTERS and a associated claim regarding to identifying the majority of element in the stream.

Algorithm 1: Heavy Hitters

Input: Frequency threshold θ, stream length m, a stream sequence $\{i_t\}_{t=1}^m$

Output: k distinct candidates elements: $\{x_j\}_{j=1}^k$

$k = \lceil \frac{1}{\theta} \rceil - 1$;
Init k pairs of tuples: $Q := \{(id_j = \perp, count_j = 0)\}_{j=1}^k$;
Let $ID = \textbf{KEY}(Q)$;
for $t = 1, \ldots m$ **do**
    if $i_t \in Q$ **then**
        $count_{i_t} = count_{i_t} + 1$;
    else if $\exists j \in [k]$; $id_j = \perp$ **then**
        $id_j = i_t$;
        $count_j = 1$;
    else
        for $j = 1, \ldots k$ **do**
            $count_j = count_j - 1$;
            if $count_j = 0$ **then**
                $id_j = \perp$;
                $count_j = 0$;
return KEY(Q);

20.2.1 Relation to selecting the majority element

1. the potential candidate finding the majority of the element is equivalent to output of HEAVY HITTER with $\theta = \frac{1}{2}$.
2. $\exists$ a majority element $x \in U$ then at the end, **HEAVY HITTER** $\left(\frac{1}{2}, m, \{i_t\}_{t=1}^m\right) = x$.

The table below illustrates the algorithm for selecting the majority element:

$\{i_t\}_{t=1}^m$	a	b	c	c	a	a	b	c	b	b	d	b	c	d	b	b	c
major$_{id}$	a	$\perp$	a	$\perp$	a	a	a	$\perp$	b	b	b	b	b	$\perp$	b	b	b
count	1	0	1	0	1	2	1	0	1	2	1	2	1	0	1	2	1

Table 1: HEAVY HITTER simulation for finding the majority

Note: the output b here is not the majority, since there is not a majority in the streaming.

20.3 Hashing

This algorithm is essentially the same algorithm as above, except that instead of mapping each streaming item i_t to a unique location, a hash function h is introduced so that $j = h(i_t), j \in [k]$ to preserve the storage space. The algorithm is described in the following:

Algorithm 2: Hashing

Input: Frequency threshold θ, stream length m, a stream sequence $\{i_t\}_{t=1}^{m}$, a hashing function h

Output: k distinct candidates elements: $\{x_j\}_{j=1}^{k}$

$k = \lceil \frac{1}{\theta} \rceil - 1$;

Init k pairs of tuples: $Q := \{(id_j = \perp, count_j = 0)\}_{j=1}^{k}$;

Let $ID = \mathbf{KEY}(Q)$;

for $t = 1, \ldots m$ **do**

  if $h(i_t) \in Q$ **then**

    $count_{h(i_t)} = count_{h(i_t)} + 1$;

  else if $\exists j \in [k]$; $id_j = \perp$ **then**

    $id_j = h(i_t)$;

    $count_j = 1$;

  else

    for $j = 1, \ldots k$ **do**

      $count_j = count_j - 1$;

      if $count_j = 0$ **then**

        $id_j = \perp$;

        $count_j = 0$;

return KEY(Q);

However, some hashing collisions would affect the performance. Thus, our next step is to analyze such effects on the performance of the algorithm.

First, treat the *count* associated with the output of hash function as an random variable. For a fixed element $x \in U$ and its hashing value $j = h(x)$, we would like to analyze the property of the random variable of $count_j$:

$$count_j = f(x) + \sum_{y \in U; y \neq x} \mathbb{1}\left(h(y) = j\right) * f(y)$$

Its expectation is:

$$\begin{aligned}
\mathbb{E}[count_j] &= f(x) + \sum_{y \in U; y \neq x} f(y) * \mathbb{P}_h\left(h(y) = j\right) \\
&= f(x) + \frac{1}{k} \sum_{y \in U; y \neq x} f(y) \\
&\leq f(x) + \frac{1}{k} \underbrace{\sum_{y \in U} f(y)}_{= \text{m}} \\
&\overset{k = \frac{1}{\epsilon}\frac{1}{\theta}}{\leq} f(x) + \epsilon\theta m
\end{aligned}$$

20.3.1 COUNT-MIN SKETCH

Alternatively, instead of using single hash function h, we uses a number of hash functions, with size P and for each element $x \in U$. Using $\min_{p\in[P]}(h_p(x))$ as aggregation function to estimate the quantity $f(x)$. This algorithm is COUNT-MIN SKETCH algorithm, stated as follows (augmented on previous algorithm):

Algorithm 3: COUNT-MIN SKETCH

Input: Frequency threshold θ, stream length m, a stream sequence $\{i_t\}_{t=1}^m$, a hashing function set $\{h_p\}_{p=1}^P$

Output: k distinct candidates elements: $\{x_j\}_{j=1}^k$

```
/* Updating the count at step t                                    */
for p ∈ [P] do
    j = h_p(i_t);
    count_{j,p} = count_{j,p} + 1;
/* After the streaming terminates, for each element x ∈ U, we take the
   minimum count over P hash functions                             */
for x ∈ U do
    return min_p count_{h_p(x),p}
```

Next, we analyze the estimator $\min_p count_{h_p(x),p}$ for $x \in U$ using probability bounding technique.

First set $k = 2\frac{1}{\epsilon}\frac{1}{\theta}$, Let $X_p = count_{jp} - f(x)$, $\mathbb{E}[X_p] = \frac{1}{2}\epsilon\theta m$.

By Markov theorem, $\mathbb{P}[X_p \geq \epsilon\theta m] \leq \frac{1}{2}$ Therefore,

$$\mathbb{P}\left(\min_p X_p \geq \epsilon\theta m\right) = \prod_{p\in[P]} \mathbb{P}[X_p \geq \epsilon\theta m]$$

$$\leq \frac{1}{2^P} := \delta$$

Therefore, with probability $1 - \delta$; for $p \geq \log(\frac{1}{\delta})$;

$$f(x) \leq \left(\min_p X_p\right) \leq f(x) + \epsilon\theta m$$

Note the space complexity is $O(kP) = O\left(\frac{1}{\epsilon}\frac{1}{\theta}\log\frac{1}{\delta}\right)$.

20.3.2 COUNT SKETCH

Notice the algorithm introduced provides an estimator that is biased, the following augmentation using a randomized updating trick to helps address this problem. Furthermore, we bound the variance of proposed estimator.

The augmented algorithm states as follows:

Algorithm 4: COUNT SKETCH

Input: Frequency threshold θ, stream length m, a stream sequence $\{i_t\}_{t=1}^m$, a hashing function set $\{h_p(x)\}_{p=1}^P$ with a associated random function $\sigma_p(x) \in \{-1, 1\}$ with equal probability.

Output: k distinct candidates elements: $\{x_j\}_{j=1}^k$

`/* Updating the count at step t */`

for $p \in [P]$ **do**

  $j = h_p(i_t)$;

  $count_{j,p} = count_{j,p} + \sigma_p(i_t)$;

`/* After the streaming terminates, for each element` $x \in U$`, we aggregate the count over` P `hash functions by taking the empirical average instead */`

for $x \in U$ **do**

  return $average\left(\sigma_p(x) * count_{h_p(x),p}\right)$

Therefore, we have:

$$\sigma_p(x) count_{j,p} = f(x) + \sum_{y \in U; y \neq x} \mathbb{1}\left(h(y) = j\right) * f(y)$$

$$\mathbb{E}\left[\sigma_p(x) count_{j,p}\right] = f(x) + \sum_{y \in U; y \neq x} f(y) * \mathbb{P}[h_p(y) = j] * \mathbb{E}[\sigma_p(x)\sigma_p(y)]$$

we assume pairwise independence for $x, y \in U$

$$= f(x) + \sigma_p(x) \frac{1}{k} \sum_{y \in U; y \neq x} f(y) \mathbb{E}[\sigma_p(y)]$$

$$= f(x), \text{ as } \mathbb{E}[\sigma_p(y) = 0]$$

Thus, we showed that $\sigma_p(x) count_{j,p}$, or $average(\sigma_p(x) count_{j,p})$ is an unbiased estimator of $f(x)$. Further, we analyze its variance.

Let $X_p = \sigma_p(x) count_{j,p} - f(x)$, then $\mathbb{E}[X_p] = 0$ and

$$Var[X_p] = \mathbb{E}[X^2] = \mathbb{E}\left[\left(\sum_{y \in U; y \neq x} f(y)\sigma_p(x)\sigma_p(y)\mathbb{1}\left(h_p(y) = x\right)\right)^2\right]$$

$$= \sum_{y \in U; y \neq x} f^2(y)\mathbb{P}(h_p(y) = x) + \underbrace{2 \sum_{y, y' \in U; y \neq y'} \mathbb{E}\left[f(y)f(y')\sigma_p(y)\sigma_p(y')\mathbb{1}(h_p(y) = x)\mathbb{1}(h_p(y') = x)\right]}_{0}$$

$$= \frac{1}{k} \sum_{y \in U; y \neq x} f^2(y)$$

$$\leq \frac{m^2}{k}, \qquad \text{since} \sum_{y \in U, y \neq x} f(y) \leq \sum_{y \in U} f(y) = m$$