Streaming Algorithms

Stream : A = <a1,a2,...,am>
    ai in [n] size log n
Compute f(A) in poly(log m, log n) space
    "one pass"

Let f_j = |{a_i in A | a_i = j}|
F_1 = sum_j f_j  = m == total count

Goal: Find all j s.t. f_j > phi m
    phi = 1/k = eps

\[ f_j - \epsilon m \leq \hat{f}_j \leq f_j \]  Misra-Greis [1985]
\[ f_j \leq \hat{f}_j \leq f_j + \epsilon m \]  Count-Min [Cormode + Muthukrishnan '05]

-------------------------
FP-MAJORITY: if some f_j > m/2, output j
    else,              output anything

How good w/ O(log m + log n)  (one counter c + one location l)?

FP-MAJORITY:
for (a_i \in A)
    if (a_i = l) c += 1
    else c -= 1
    if (c <= 0)  c = 1, l = a_i
return l

Analysis: if f_j > m/2, then
    if (l != j) then c decremented at most < m/2 times, but c > m/2
    if (l == j) can be decremented < m/2, but is incremented > m/2
    if f_j < m/2 for all j, then any answer ok.

-------------------------
k-FREQUENCY-ESTIMATION: Build data structure S.
For any $j$ in $[n]$, $\hat{f}_j = S(j)$ s.t.
\[ f_j - \frac{m}{k} \leq \hat{f}_j \leq f_j \]

aka eps-approximate phi-HEAVY-HITTERS:
- Return all $f_j$ s.t. $f_j > \phi m$
- Return no $f_j$ s.t. $f_j < \phi m - \epsilon m$
- (any $f_j$ s.t. $\phi m - \epsilon m < f_j < \phi m$ is ok)

-------------------------

Misra-Gries Algorithm [Misra-Gries '82]

Solves $k$-FREQUENCY-ESTIMATION in $O(k(\log m + \log n))$ space.

Let $C$ be array of $k$ counters $C[1], C[2], \ldots, C[k]$
Let $L$ be array of $k$ locations $L[1], L[2], \ldots, L[k]$

Set all $C = 0$
Set all $L = X$

for $(a_i \in A)$
  if $(a_i \in L)$  
    $C[j] += 1$
  else
    if $(|L| < k)$
      $C[j] = 1$
      $L[j] = a_i$
    else
      $C[j] -= 1$ for all $j$ in $[k]$

for $(j \in [k])$
  if $(C[j] \leq 0)$ set $L[j] = X$

On query $q$ in $[n]$
  if $(q \in L \{L[j]=q\})$ return $\hat{f}_q = C[j]$
  else return $\hat{f}_q = 0$

Analysis

A counter $C[j]$ representing $L[j] = q$ is only incremented if $a_i = q$

$\hat{f}_q \leq f_q$
If a counter $C[j]$ representing $L[j] = q$ is decremented, then $k - 1$ other counters are also decremented. This happens at most $m/k$ times. A counter $C[j]$ representing $L[j] = q$ is decremented at most $m/k$ times.

$$f_q - m/k \leq \hat{f}_q$$

How do we get an additive $\varepsilon$-approximate FREQUENCY-ESTIMATION? i.e. return $\hat{f}_q$ s.t.

$$|f_q - \hat{f}_q| \leq \varepsilon m$$

Set $k = 2/\varepsilon$, return $C[j] + (m/k)/2$

Space $O((1/\varepsilon)(\log m + \log n))$

Also:
eps-approximate phi-HEAVY-HITTERS for any phi > m*eps in space $O((1/\varepsilon)(\log m + \log n))$

------------------------------------

COUNT MIN Sketch
------------------------------------

t independent hash functions $\{h_1, \ldots, h_t\}$
each $h_i : [n] \rightarrow [k]$

2-d array of counters:
h_1 -> $[C_{1,1}] [C_{1,2}] \ldots [C_{1,k}]$
h_2 -> $[C_{2,1}] [C_{2,2}] \ldots [C_{2,k}]$
... ... ...
h_t -> $[C_{t,1}] [C_{t,2}] \ldots [C_{t,k}]$

for each $a$ in $A$ -> increment $C_{i,h_i(a)}$ for $i$ in $[t]$.

$\hat{f}_a = \min_{i \in [t]} C_{i,h_i(a)}$

Set $t = \log(1/\delta)$
Set $k = 2/\varepsilon$

---------------------------

Clearly $f_a \leq \hat{f}_a$

$\hat{f}_a \leq f_a + W$. What is $W$?
One hash function $h_i$. Adds to $W$ when there is a collision $h_i(a) = h_i(j)$. $wp \ 1/k$

Random variable $Y_{i,j}$

$Y_{i,j} = \{f_j \ wp \ 1/k, 0 \ wp \ 1-1/k\}$

$E[Y_{i,j}] = f_j / k$

Random variable $X_i = \sum_{j \in [n], j!=a} Y_{i,j}$

$E[X_i] = E[\sum_j Y_{i,j}] = \sum_j f_j / k = F_1 / k = \epsilon * F_1 / 2$

Markov Inequality

$X$ a rv and $a>0$

$Pr[|X| >= a] <= E[|X|] / a$

$X_i > 0$ so $|X_i| = X_i$

setting $a = \epsilon F_1$ then

$E[|X_i|] / a = (\epsilon F_1 / 2) / (\epsilon F_1) = 1/2$

$Pr[X_i >= \epsilon F_1] <= 1/2$

-----

Now for $t$ *independent* hash functions:

$Pr[\hat{f}_a - f_a >= \epsilon F_1]$

$= Pr[min_i X_i >= \epsilon F_1]$

$= Pr[\forall_{i \in [t]} (X_i >= \epsilon F_1)]$

$= \prod_{i \in [t]} Pr[X_i >= \epsilon F_1]$

$<= 1/2^t$

$= \delta$ (since $t = \log(1/\delta)$)

Hence:

$f_a <= \hat{f}_a <= f_a + \epsilon F_1$

- first inequality always holds
- second inequality holds $wp > 1-\delta$

-----

Space:

each of $k*t$ counters requires $\log m$ space

$O(k*t*\log m)$

Store $t$ hash functions: $\log n$ each

$O((k \log m + \log n)*t) = O((1/\epsilon) \log m + \log n) \log (1/\delta))$

--------------------------

turnstile model: add or subtract (as long as is there)