Streaming

Inge Li Gørtz

Streaming model (one-pass)

- Stream.
    - Elements $a_1, a_2, \ldots, a_m$ from the universe $[n] = \{1,2,\ldots,n\}$.
    - Elements arrive one by one.
    - Must process element a_i before we see a_{i+1}.

4	3	2	4	4	3	1	1	2	5	9	7	1	3

- Space. Measured in bits.
- Goal. Small space (sublinear/polylogarithmic).
- Example. What can we do in $O(\log n + \log m)$ space?

Frequent elements

Frequent elements

- Heavy Hitters Problem. Find all elements *i* that occurs more than *m/k* times for some fixed *k*.
- Example. Return all elements that occur more than 21/3 times = 7.

4	4	1	2	4	4	3	1	1	2	5	9	7	4	1	3	4	1	4	4	1

Frequent elements

- Heavy Hitters Problem. Find all elements *i* that occurs more than *m/k* times for some fixed *k*.
- Example. Return all elements that occur more than 21/3 times = 7.

4	4	1	2	4	4	3	1	1	2	5	9	7	4	1	3	4	1	4	4	1

Frequent elements

- Heavy Hitters Problem. Find all elements *i* that occurs more than *m/k* times for some fixed *k*.
- Example. Return all elements that occur more than 21/3 times = 7.

4	4	1	2	4	4	3	1	1	2	5	9	7	4	1	3	4	1	4	4	1

- Bad news. Need $\Omega(n)$ space for one-pass algorithm.
- Good news.
  - Can estimate the frequency.
  - Can do better if we allow one-sided error:
    - Output all elements that occur more than m/k times.
    - Might also output other elements.

Frequent elements

- Heavy Hitters Problem. Find all elements *i* that occurs more than *m/k* times for some fixed *k*.
- Example. Return all elements that occur more than 21/3 times = 7.

4	4	1	2	4	4	3	1	1	2	5	9	7	4	1	3	4	1	4	4	1

- Misra-Gries.

```
Keep k-1 counters in an associative array A.
while (stream is not empty) do
    If j ∈ keys(A) then
        A[j] ← A[j] + 1
    else if |keys(A)| < k − 1 then
        A[j] ← 1
    else
        Decrement all counters by 1.
        Remove all elements with counter 0.
Output all elements in keys(A)
```

- Space. $O(k \cdot (\log n + \log m))$.

Frequent elements

- Example. k = 3.

4	4	1	2	4	4	3	1	1	2	5	9	7	4	1	3	4	1	4	4	1

counter 1: 4 , 2

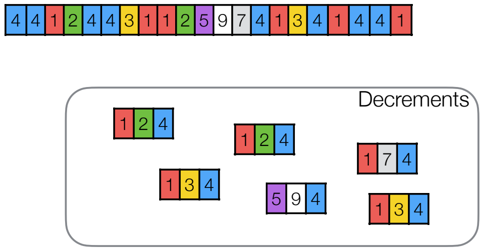
counter 2: 1 , 1

```
Keep k-1 counters in an associative array A.
while (stream is not empty) do
    If j ∈ keys(A) then
        A[j] ← A[j] + 1
    else if |keys(A)| < k − 1 then
        A[j] ← 1
    else
        Decrement all counters by 1.
        Remove all elements with counter 0.
Output all elements in keys(A)
```

Misra-Gries Analysis

- Lemma. Any item with frequency more than m/k is in A by the end of the algorithm.
- Lemma. Let $\hat{f}_i$ be the estimate of the frequency of element i. Then

$$f_i - \frac{m}{k} \le \hat{f}_i \le f_i \ .$$



Reservoir Sampling

Reservoir Sampling

- Algorithm.

put the first k elements into a "reservoir" $R = \{r_1, r_2, \ldots, r_k\}$.
for $i > k$ until the stream is empty do
  with probability k/i replace a random entry of R with a_i
Return R.

- Claim. For all $t \ge i$, $P[a_i \in R_t] = k/t$, where R_t denotes the reservoir after time t.
- Proof. By induction on t.
  - Base case: $t = k$ (assume stream has length at least k).

$$P[x_i \in R_t] = 1 = k/t \,.$$

Reservoir Sampling

- Algorithm.

put the first k elements into a "reservoir" $R = \{r_1, r_2, \ldots, r_k\}$.
for $i > k$ until the stream is empty do
  with probability k/i replace a random entry of R with a_i
Return R.

- Claim. For all $t \ge i$, $P[a_i \in R_t] = k/t$, where R_t denotes the reservoir after time t.
- Proof. By induction on t. Assume $t > k$.
  - Case 1. $i = t$.

$$P[a_i \in R_t] = P[a_t \in R_t] = k/t \,.$$

Reservoir Sampling

- Algorithm.

> put the first k elements into a "reservoir" $R = \{r_1, r_2, \ldots, r_k\}$.
> for $i > k$ until the stream is empty do
>   with probability k/i replace a random entry of R with a_i
> Return R.

- Claim. For all $t \geq i$, $P[a_i \in R_t] = k/t$, where R_t denotes the reservoir after time t.

- Proof. By induction on t. Assume $t > k$. Case 2. $i < t$.

$$P[x_i \in R_t] = P[x_i \in R_{t-1}] \cdot P[x_i \text{ not replaced at time } t \,|\, x_i \in R_{t-1}]$$

$$= \frac{k}{t-1} \cdot P[x_i \text{ not replaced at time } t \,|\, x_i \in R_{t-1}]$$

$$= \frac{k}{t-1}\left(1 - \frac{k}{t} \cdot \frac{1}{k}\right) = \frac{k}{t}$$