| | |
|---|---|
| **CSCE 689: Special Topics in Modern Algorithms for Data Science** | Fall 2023 |

Lecture 10 — September 13, 2023

*Prof. Samson Zhou* *Scribe: Zhitong Chen*

## Frequent Items

- **Goal**: Given a set $S$ of $m$ elements from $[n]$ and a parameter $k$, output the items from $[n]$ that have frequency at least $\frac{m}{k}$.
- How many items can be returned?

**The Answer is**: at most $k$ coordinates with frequency at least $\frac{m}{k}$. **Why?** Assuming that more items are returned than $k$, then the frequency of each item is at least $\frac{m}{k}$, which means that the total number of elements will be more than $m$, which contradicts the fact that we only have $m$ elements.

- For $k = 20$, want items that are at least 5% of the stream.

**The Answer is**: First, consider that we are looking for items with a frequency of at least $\frac{m}{k}$, and when $k = 20$, we are looking for items with a frequency of at least $\frac{m}{20}$. Converting this frequency to a percentage: $\frac{(m/20)}{m}$. Simplifying this expression, we get $\frac{1}{20}$, which is 5%, so for $k = 20$, we want to find those items that make up at least 5% of the data stream.

## Majority

- **Goal**: Given a set $S$ of $m$ elements from $[n]$ and a parameter $k = 2$, output the items from $[n]$ that have frequency at least $\frac{m}{2}$.

**Explanation**: When you have a set $S$ containing $m$ elements and they come from $[n]$ (which might be a larger range or set), and given the parameter $k = 2$, the goal is to find those terms that have a frequency of at least $\frac{m}{2}$ among those $m$ elements. Since $k$ has been defined as 2, we would like to find terms that occur at least half as often as the whole stream.

- Find the item that forms the majority of the stream

**Algorithm (Boyer-Moore majority vote algorithm)**: Initialize item $V = \perp$ with count $c = 0$. For updates $i = 1, \ldots, m$: If $c = 0$, set $V = x_i$ and $c = 1$. Otherwise if $V = x_i$, increment $c$. Otherwise if $V \neq x_i$ and $c > 0$, decrement $c$.

**Intuition**: Initialize $V = \perp$ and counter $c = 0$. If $x_1$ is not the majority item, it must be deleted at some time $T$. At time $T$, the stream will have consumed $\frac{T}{2}$ instances of $x_1$, so that the majority item of the stream must have only appeared at most $\frac{T}{2}$ times. Thus of the remaining $\frac{m}{2} - \frac{T}{2}$ updates, the majority item of the entire stream remains the majority over the rest of the stream.

## Misra-Gries Algorithm

- **Goal**: Given a set $S$ of $m$ elements from $[n]$ and a parameter $k$, output the items from $[n]$ that have frequency at least $\frac{m}{k}$.

**Algorithm**: Initialize $k$ items $V_1, ..., V_k$ with count $c_1, ..., c_k = 0$. For updates $i = 1, ..., m$: If $V_t = x_i$ for some $t$, increase counter $c_t$ by setting $c_t = c_t + 1$. Else if $c_t = 0$ for some $t$, set $V_t = x_i$. Else decrease all counters $c_t = c_t - 1$.

**Claim**: At the end of the stream of length $m$, we report all items with frequency at least $\frac{m}{k}$.

**Intuition**: If there are k coordinates with frequency $\frac{m}{k}$, they will all be tracked and reported, since we have $k$ counters. If there are $\frac{m}{2}$ coordinates with frequency at least $\frac{m}{k}$, we still have $\frac{k}{2}$ counters for the remaining $\frac{m}{2}$, updates Will have at most $\frac{m}{k}$ decrement operations, which is small enough so that frequent items are still stored.

However, the Misra-Gries algorithm has some drawbacks. Misra-Gires may return false positives, like the items that are not frequent.

In fact, no algorithm using $o(n)$ space can output, ONLY the items with frequency at least $\frac{n}{k}$.

**Intuition**: Hard to decide whether coordinate has frequency $\frac{n}{k}$ or $\frac{n}{k} - 1$.

**Example**: Suppose $n' = \Theta(n)$ items appear either once or never, e.g., $x_1 = 2$, $x_2 = 5$, $x_3 = 4$, $x_4 = 7$, $x_5 = 1$, $x_6 = 9$, $\ldots$ Then suppose a single random item appears $\frac{n}{k} - 1$ times, e.g., $x_{\frac{n}{k}+1} = a$, $x_{\frac{n}{k}+2} = a$, ..., $x_n = a$. Then $a$ appears $\frac{n}{k}$ if and only if it appears in the first $n'$ items. However, this requires storing the entire set of $\Theta(n)$ items.

## $(\varepsilon, k)$-Frequent Items Problem

- **Goal**: Given a set $S$ of $m$ elements from $[n]$, an accuracy parameter $\varepsilon \in (0, 1)$ , and a parameter $k$, output a list that includes:
  - The items from $[n]$ that have frequency at least $\frac{m}{k}$
  - No items with frequency less than $(1 - \varepsilon)\frac{m}{k}$

## Misra-Gries for $(\varepsilon, k)$-Frequent Items Problem

**Algorithm**: Set $r = [\frac{k}{\varepsilon}]$ and initialize $r$ items $V_1, ..., V_r$ with count $c_1, ..., c_r = 0$. For updates $i = 1, \ldots, m$: If $V_t = x_i$ for some $t \in [r]$, increment counter $c_t$, i.e., $c_t = c_t + 1$. Else if $c_t = 0$ for some $t \in [r]$, set $V_t = x_i$. Else decrement all counters $c_j$, i.e., $c_t = c_t - 1$ for all $t \in [r]$.

**Claim**: For all estimated frequencies $\hat{f}_i$ by Misra-Gries, we have

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

In particular, if $f_i \geq \frac{m}{k}$, then

$$\hat{f}_i \geq f_i - \frac{\varepsilon m}{k}$$

and if $f_i < (1-\varepsilon) \cdot \frac{m}{k}$, then

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

Thus if we return coordinates $V_t$ with $c_t \geq (1-\varepsilon) \cdot \frac{m}{k}$ then:

- $i$ with $f_i \geq \frac{m}{k}$ will be returned
- No $i$ with $f_i < (1-\varepsilon) \cdot \frac{m}{k}$ will be returned

**Summary**: Misra-Gries can be used to solve the $(\varepsilon, k)$-frequent items problems. It is a deterministic algorithm that uses $O\left(\frac{k}{\varepsilon} \log n\right)$ bits of space and it always underestimates the true frequency.