Lab 8

Streaming - Majority element and F0 estimation

Majority element (heavy hitters algorithm)

Assume the length of a data stream is n, how to find if there is a element that appears more than n/2 times with constant space? How many passes you need to make?

Majority element

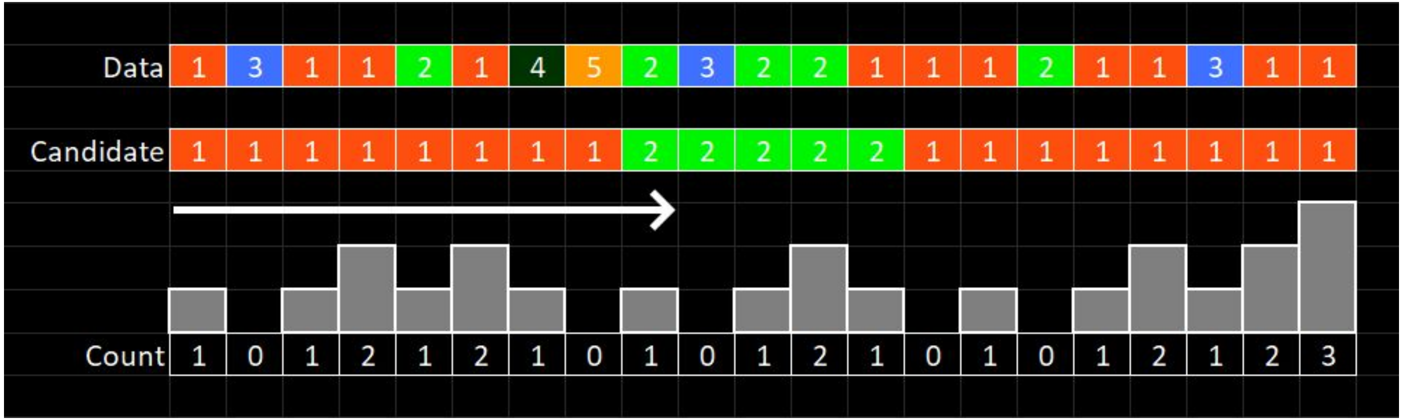
Assume the length of data sequence is n, how to find if there is a element that appears more than n/2 times with constant space?

Hint:

1. Assume elements are all integers in Python (4 bytes each). Only 8 bytes are needed.

Majority element

- Name key-value pair KV=(KV[0], KV[1])
- For each element e in the stream:
  - If the key-value pair is empty, set it to be (e, 1)
  - If KV not empty, and e = KV[0], then set KV[1] += 1
  - If KV not empty, and e != KV[0], then KV[1] -= 1. Empty KV if KV[1]=0.
- Go through the stream again to check the frequency of KV[0].



Distinct elements

Lower bound on memory for exact deterministic algorithm

- Consider a sequence of m+1 elements.
- There are $2^m - 1$ possible subsets of elements for first m elements.
- To determine the exact number of distinct elements in the sequence, we need at least m bits of memory.
- If only m-1 bits are used, then the memory can only have 2^{m-1} states.
    - Two different subsets will share one state, which leads to incorrect answer.

Can we use sampling to approximate F0?

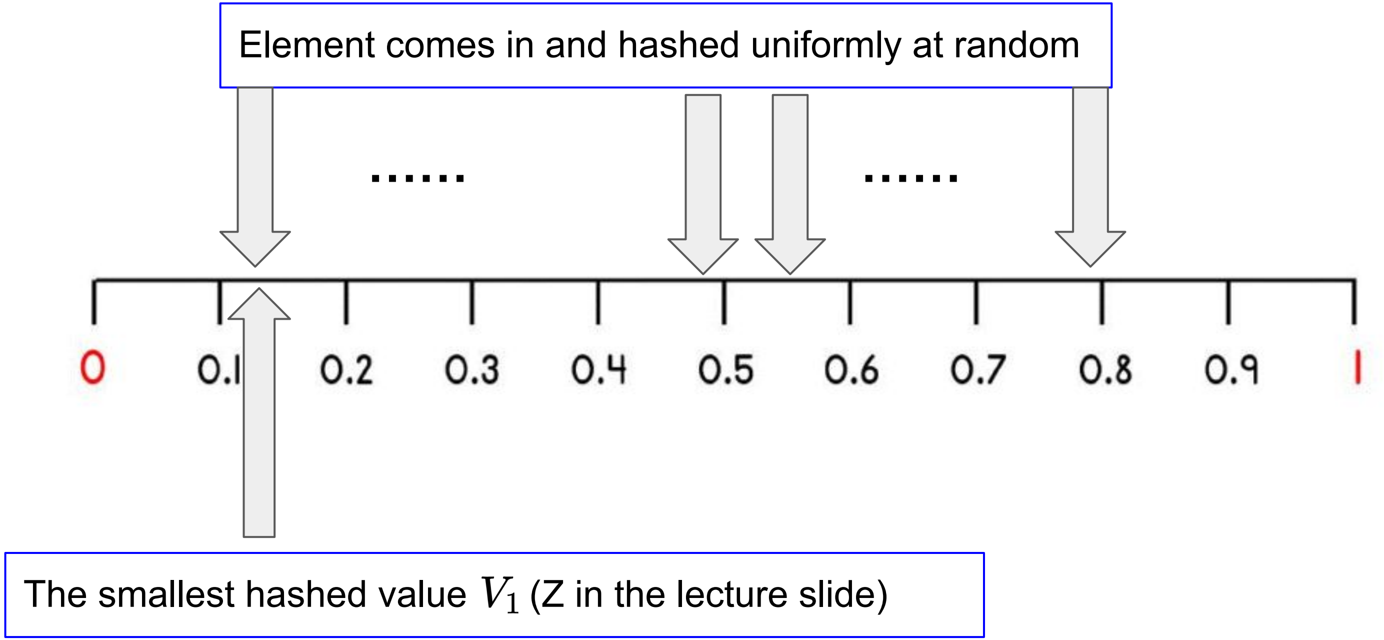
In general, the answer is no.

Bad example: Assume the length of the stream is m.

1. The stream with m-1 0's and one 1 has F0=2.
2. The stream with m/2 0's and m/2 1's also has F0=2.

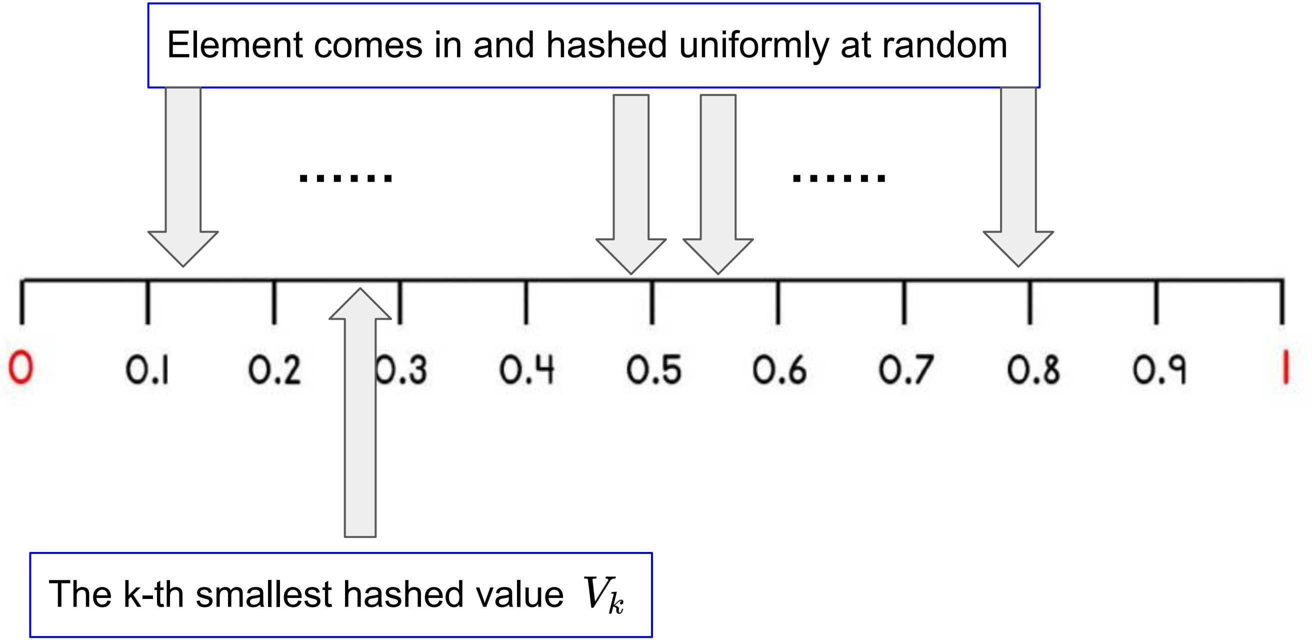
Sampling cannot catch the minority with high probability, unless all elements appears with similar frequencies.

Distinct element estimation using k-th min (HW)



$$\mathbb{E}[\mathbb{Z}] = \int_0^1 \Pr(Z > t)dt = \int_0^1 \Pr\left(X_1 > t\right)^n = \int_0^1 (1-t)^n dt = \frac{1}{n+1}$$

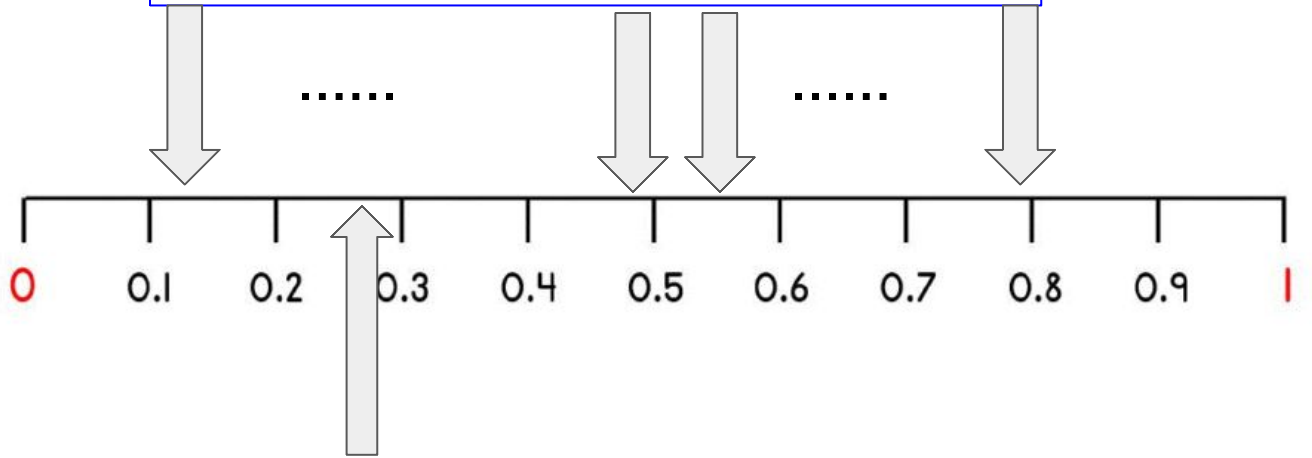
Distinct element estimation using k-th min (HW)



$$\Pr[V_k \leq x] = \Pr[\text{at least k observations are } \leq \text{ x}] = \sum_{l=k}^{n} \binom{n}{l} x^l (1-x)^{n-l}$$

Distinct element estimation using k-th min (HW)

Element comes in and hashed uniformly at random



The k-th smallest hashed value V_k

$$\frac{d}{dx}\sum_{l=k}^{n}\binom{n}{l}x^l(1-x)^{n-l} = \sum_{l=k}^{n}\binom{n}{l}\left(lx^{l-1}(1-x)^{n-l} - x^l(n-l)(1-x)^{n-l-1}\right)$$

Compute on black board

Distinct element estimation using k-th min (HW)

$$\frac{d}{dx}\sum_{l=k}^{n}\binom{n}{l}x^l(1-x)^{n-l} = \sum_{l=k}^{n}\binom{n}{l}\left(lx^{l-1}(1-x)^{n-l} - x^l(n-l)(1-x)^{n-l-1}\right)$$

$$= n\binom{n-1}{k-1}x^{k-1}(1-x)^{(n-1)-(k-1)}$$

This is the pdf of beta distribution!

$$E_{X\sim \mathrm{Beta}(\alpha,\,\beta)}[X] = \frac{\alpha}{\alpha+\beta}$$