CS5234

Jia Cheng

August 2022

Lemma Let m be a variable. $o(f_1(m)) + o(f_2(m)) = o((f_1 + f_2)(m))$.

Proof. The variable m is implicit for simplicity. Let $g_i = o(f_i), i \in \{1, 2\}$. Then

$$\frac{g_1 + g_2}{f_1 + f_2} = \frac{g_1}{f_1 + f_2} + \frac{g_2}{f_1 + f_2} \leq \frac{g_1}{f_1} + \frac{g_2}{f_2} \to 0 \qquad \text{as } m \to \infty \tag{1}$$

which proves the formula. As usual in algorithmic analysis, we assume that all functions are non-negative.

Proposition Let m be the number of elements. There does not exist a deterministic algorithm that is $o(m)$ for finding the median.

Proof. Suppose there does exist such a comparison based algorithm A. We can then use A as follows. Denote the input as `input`.

1. Create empty array `arr` of length m.
2. Set `arr[rank(m/2)]` $\leftarrow$ `median(input[1..m])`.
3. Repeat this for the subarrays `input[1..rank(m/2)-1]`, `input[rank(m/2)+1..m]` and so on.

The runtime will then be $T(m) = 2T(m/2) + o(m)$, which expands to $T(m) = o(m \log m)$, which is impossible for a comparison based algorithm.

This doesn't reach a contradiction for a non-comparison based algorithm.

Proof. My second idea is to consider the idea that it is necessary for a deterministic algorithm to at least read every single input element before making a decision on the median, and this reading would in itself take m steps. Suppose some deterministic algo A attempts to choose the median in fewer than m steps, then there exists some element that the algo has not read. We can then manipulate this unread element, without touching the other elements to obtain an indistinguishable input which has a different median.

Proposition Markov's inequality

Let X be a non-negative r.v. Then $\forall k$

$$E[X] \geq kP(X \geq k) \tag{2}$$

In particular, when $k > 0$

$$P(X \geq k) \leq \frac{E[X]}{k} \tag{3}$$

Proposition Chebyshev's inequality

Let X be a r.v. Then $\forall k \geq 0$,

$$P(|X - E[X]| \geq k) \leq \frac{Var(X)}{k^2} \tag{4}$$

One possible derivation is by applying Markov's inequality to $(X - E[X])^2$.

Proposition Chernoff Bound (Simplified)

Let X be a sum of independent r.v. $X_i \sim Bernoulli(p_i)$. (Note that this also implies X is non-negative.) Then $\forall \epsilon \geq 0$,

- $P(X \geq (1+\epsilon)E[X]) \leq e^{-\frac{\epsilon^2 E[X]}{2+\epsilon}}$
- $P(X \leq (1-\epsilon)E[X]) \leq e^{-\frac{\epsilon^2 E[X]}{2}} \leq e^{-\frac{\epsilon^2 E[X]}{3}}$

In particular, when $\epsilon \leq 1$,

$$P(X \geq (1+\epsilon)E[X]) \leq e^{-\frac{\epsilon^2 E[X]}{3}} \tag{5}$$

Discussion Process 1 analysis.

Discussion Process 2 analysis.

Discussion Process 3 analysis.

Discussion We discuss the properties of Process 1, 2 and 3.

All 3 processes involve sampling, but processes 1 and 2 do not provide an computational way to conduct sampling. When the dataset m is large, we cannot store the entire thing. When the data is a stream, we don't know m.

Process 3 has the advantage of being actually implementable, because it

- Does not require more than $O(t \log n)$ bits storage (when n is the numerical size of the stream data elements)
- Can sample without knowing m using reservoir sampling

Problem Design an extension to the uniform sampling algorithm of 1 element that produces a sample of size t.

Solution. Denote m (unknown) as the size of the stream. Denote the i-th stream element as x_i. Assume that $t \leq m$.

1. Let `arr[1..t]` be an array of size t.
2. Let i be an iterator variable from 1 to m.
   (a) If $1 \leq i \leq t$, then `arr[i]` $\leftarrow x_i$
   (b) If $i > t$, then with probability $\frac{t}{i}$, x_i replaces a uniformly chosen element of the array. Otherwise, with remaining probability $1 - \frac{t}{i}$, no replacement is done.

We claim that this sampling method produces a uniform sample of size t. We shall prove this by induction, using the statement $Q(i)$ for $i \geq t$: At the i-th step of the loop, the probability of $x_j, 1 \leq j \leq i$ being in the array is $\frac{t}{i}$.

The base case $Q(t)$ is clearly true since $x_j, j \leq t$ must be in the array.

Suppose $Q(k)$, then x_{k+1} will go into the array with probability $\frac{t}{k+1}$. Consider some element $x_j, j < k+1$. Then

$$P(x_j \in arr \text{ at step k+1}) = P(x_j \in arr \text{ at step k+1} \mid x_j \in arr \text{ at step k}) P(x_j \in arr \text{ at step k}) \tag{6}$$

The right term of the product is by induction hypothesis $\frac{t}{k}$, whereas the left term is given by $\left(1 - \frac{t}{k}\right) + \frac{t}{k} \cdot \frac{1}{t}$, the product evaluates to $\frac{t}{k+1}$, as desired.

We have completed our induction. □