Mohammad Sadegh Akbari
msa1376.sa@gmail.com

HW-2

Sublinear Algorithms,Random Walk

Sublinear Algorithms:

To gain an understanding of sublinear time algorithms, you can refer to the following two videos: *video 1*, *video 2*

1.

حالت ضربی نامساوی چرنوف را به صورت زیر در نظر بگیرید.

لم ۱ اگر $X_1, \ldots, X_n$ متغیرهای تصادفی دودویی باشند و $X = X_1 + \cdots + X_n$، آنگاه برای هر $0 \leq \epsilon \leq 1$ داریم

$$\Pr[|X - E[X]| \geq \epsilon E[X]] < 2 \exp\left(\frac{-\epsilon^2 E[X]}{3}\right)$$

با استفاده از این لم، الگوریتمی برای مساله‌ی پیش رو ارائه دهید. آرایه‌ی n عضوی A متشکل از صفر و یک است. می‌خواهیم تقریبی بررسی کنیم که آیا تعداد یک‌ها از k بیشتر است یا خیر. به صورت دقیق‌تر، اگر تعداد یک‌ها بیشتر یا مساوی $(1+\epsilon)k$ است باید جواب آری برگردانیم، اگر تعداد یک‌ها کمتر مساوی $(1-\epsilon)k$ است جواب خیر و در غیر این صورت جوابی برنگردانیم. الگوریتمی ارائه دهید که با $O(n/k.\epsilon^{-2}.\log \delta^{-1})$ بار دسترسی، جوابی با احتمال درستی $1-\delta$ بازگرداند.

2.

Definition: a list of size n is ε-close to sorted if can delete at most εn values to make it sorted. Otherwise, ε-far.

Given a list $y_1, \cdots, y_n$ of integers, our goal is to output whether or not the list is sorted:

• If the list is sorted, that is $y_1 \le y_2 \le \cdots \le y_n$ then output PASS.

• If the list is ε-far from sorted in Hamming distance, that is > εn elements need to be changed to make the list sorted, then output FAIL with probability ≥ *3/4*.

(hint: you can use below algorithm)

Input : List $y_1, y_2, \cdots, y_n$, and value ϵ
Output: pass or fail
for $O(1/\epsilon)$ *random indices* i *in* $[1, n]$ **do**
    Set $z \leftarrow y_i$;
    Do binary search on $y_1, y_2, \cdots, y_n$ for z;
    if *see inconsistency (left is bigger, right is smaller, or* z *not found)* **then**
        Fail and halt
Pass

3.

In class, you see an MST approximation algorithm for graphs in which the weights on each edge were integers in the set $\{1, \ldots, w\}$.

Show that one can get an approximation algorithm when the weights can be any value in the range $[1, w]$. (It is okay to have a slightly worse running time in terms of $w, 1/\varepsilon$).

(Hint: Take the case where $w=2$: consider dividing the range $[1,2]$ into pieces of length ρ with endpoints in $L = \{1, 1+\rho, 1+2\rho, \ldots, 2-\rho, 2\}$. Rounding the values of the actual weights to the nearest value in L. Show that using the new weights won't change the value of the output MST by much. Show how to adapt the algorithm from class to work in this "discretized" setting.)

4.

we have a stream of n integers ranging from a to b and a given number k, ε.

consider a median-finding algorithm that selects k random samples and returns their median. Determine the value of δ such that with a probability of at least $(1-\delta)$, the median of the sample deviates from the true median by no more than $[\varepsilon * n/2]$ elements.

(hint: this problem can be simplified, without loss of generality, to randomly sampling k integers from 1 to n and comparing the sample median with n/2)

Random Walk:

5.

Given any connected graph with *n = number of vertices* and *m = number of edges*, determine a function *f(n, m, y)* that is linear in *n, m*, and *y*, such that a random walk of *f(n, m, y)* steps ensures each node is visited at least once with a probability of at least *1 - 2^(-y)*, regardless of the starting node.

(hint: start with Markov's inequality, which states that *Pr(X > w) <= E(X)/w)*

6.

Imagine you're stranded in Monte Carlo with only k dollars to your name. You have one day to earn m dollars to repay a debt—or face death. Your only option is to play a risky casino game with these rules:

1. Gain *1* dollar on winning a round.

2. Lose *1* dollar on losing a round.

3. Each round has a *50*% chance of winning or losing.

4. You can play up to m rounds tomorrow.

Can you devise an algorithm to calculate the probability of you seeing the day after tomorrow?

(hint: Construct a probability vector $\mathbf{v}_i \in \mathbb{R}^n$, where each element $\mathbf{v}_i[j]$ signifies the probability of possessing j dollars at the i-th game.Define a transition matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ that governs the evolution of $\mathbf{v}_i$, satisfying the following relationship: $\mathbf{v}_{i+1} = \mathbf{A}\mathbf{v}_i$. This approach make an algorithm with a time complexity of O($n^2 * m$))