Big Data Algorithm – Final Exam

December 15, 2017

Problem 1

Morris Counting (Proposed by Robert Morris 1977) algorithm allows the counting of a large number of elements using memory of which size is small. The algorithm make the incrementing of counts become a probabilistic event. Assuming the current counts is X_n (have read n elements), the probability of incrementing is 2^{-X_n}, and make $\hat{n} = 2^{X_n} - 1$.

Now we have two Morris counters X_n, Y_n for streams $\mathbf{X}, \mathbf{Y}$ for size n_x, n_y, we want to merge them: obtain a single counter Z_n which has the same distribution (is a Morris counter) for a stream of size $n_x + n_y$.

Consider two scenarios: (a). Morris counting applied to $\mathbf{Y}$. (b). Morris counting applied to $\mathbf{Y}$ after $\mathbf{X}$.

We want to simulate the result of (b) given $\mathbf{Y}$ (result of (a)) and $\mathbf{X}$.

1.if $X_n = 0$, Z_n =?

2.Do elements of $\mathbf{Y}$ that did not increment in (a) increment in (any corresponding run of) (b)? Why?

3.If the element that had the $i-$th increment in (a), what is the condition probability()that the element has the increment on (b)? (Now the morris counting in (b) is Z_k, and $Z_k > i$)

4.(bonus) Give the processing of merging them as detail as possible. You can write pseudocode.

(tips: you can suppose a random variable following the uniform distribution $\mathbf{U}[0, 1]$ for explanation)

Problem 2

If $\mathcal{U}$ is a universal Turing machine, for any other universal Turing machine $\mathcal{U}'$,please prove that there exists a constant $c_{\mathcal{U}'}$ such that

$$K_{\mathcal{U}}(x) \leq K_{\mathcal{U}'}(x) + c_{\mathcal{U}'}$$

for all strings $x \in \{0, 1\}^*$ and the constant does not depend on x.

Problem 3

Give n points and a number k, assume that n is more big than k. How to get k convex hulls and they don't intersect with each other?

Problem 4

About the random median finding algorithm with input size N and memory size $F(N > F)$:

- Please analyze the probability that the median is retained in the memory by the end of the algorithm. You just need to analyze under what circumstance the median is retained. You don't have to calculate the specific probability
- Consider a simple random walk model, with the probability equals to 1/2 for both walking left and right, let n1 denotes the steps taken to left and n2 to right, and N steps in total. Please calculate the mean value of n1.
- Let $S_n = n1 - n2$, please calculate the mean value of S_n and S_n^2.

Detailed deduction or calculation is required for these questions.

Problem 5

Considering LRU paging algorithm, given a request sequence σ and let b a parameter. Let $cost(A, k, \sigma)$ denote the number of page faults incurred by the paging algorithm A with a cache size of k on the page sequence σ. Defining a cache size k is bad, i.e., LRU can obtain improvement of factor 2 when augmented resource is b, if

$cost(LRU, k + b, \sigma) < \frac{1}{2} cost(LRU, k, \sigma)$

- Giving an upper bound of $cost(LRU, k, \sigma)$ when cache size is not bad.

  Note: Assuming $cost(OPT, k, \sigma)$ is already known and considering LRU algorithm with augmented resource.

- $\forall \epsilon, \delta > 0, \forall n \in N^*, \forall \sigma$, Prove: For all but a fraction of the cache sizes k in $\{1, 2, \ldots, n\}$ that are bad cache sizes, the LRU paging algorithm satisfies either:

  (1). $cost(LRU, k, \sigma) = O(\frac{1}{\delta} log_2 \frac{1}{\epsilon}) cost(OPT, k, \sigma)$; or

  (2). $cost(LRU, k, \sigma) \leq \epsilon \mid \sigma \mid$ ($\mid \sigma \mid$ is the length of σ)

  Note1: Considering intervals between bad cache sizes, i.e., the number of augmented resource b, is equal to

$$\frac{\delta n}{log_2 \epsilon^{-1}} \tag{1}$$

  Note2: When $b = \frac{\delta n}{log_2 \epsilon^{-1}}$, $\frac{2(n+b)}{b+1} = \Theta(\frac{1}{\delta} log_2 \frac{1}{\epsilon})$

Problem 6

Recall that the count distinct items algorithms in the lecture uses $GF(2^d)$ as the hash function and the property of F_0 distinct randomized items in $GF(2^d)$ to estimate F_0.

1. Give at least two advantages of using $GF(2^d)$ compared with $GF(p)$, where p is a prime.
2. Suppose there is another universal hash function, whose output is uniformly distributed between 0 and 1 for any input. Can you design another algorithm for counting distinct items using the given hash function?
3. (Bonus) Can you analyze the performance of your proposed algorithm? i.e. Can you derive the lower bound of $P(|Y - F_0| \leq \epsilon F_0)$ or $P(1/\epsilon \leq Y/F_0 \leq \epsilon)$, or something similar.

Problem 7

In the midterm exam, we have poved the Theorem 1.

Theorem 1 For an n-vertex graph G = (V, E) with m edges, two disjoint sets $C_0 \subseteq V$ and $V_0 \subseteq V$can be computed in time $O(\sqrt{n}, m)$, such that the following three properties hold.
1. Let $D \subseteq V_0$ be a vertex cover of the subgraph $G[V_0]$. Then $C := D \cup C_0$ is a vertex cover of G.
2. There is a minimum vertex cover S of G with $C_0 \subseteq S$.
3. The subgraph G[V0] has a minimum vertex cover of size at least $|V_0|/2$.

Now, please prove Theorem 1 based on Theorem 2.

Theorem 2 Let (G = (V, E), k) be an input instance of $V_{ERTEX} \quad C_{OVER}$. In time $O(k \cdot |V| + k^3)$ one can compute a reduced instance$(G\prime = (V\prime, E\prime), k\prime)$ with $|V\prime| \leq 2k$ and $k\prime \leq k$ such that G admits a vertex cover of size k iff $G\prime$ admits a vertex cover of size $k\prime$.

Problem 8

1. There is a vector A=[1,0,-2,0,3,0,-4,0,5,0,-6,0]. What is the result of $||A||_0, ||A||_1, ||A||_2, ||A||_\infty$.
2. For finite dimension vector A, prove that

$$\exists c_1, c_2 > 0, \text{ for any s and t, } c_1||A||_s \leq ||A||_t \leq c_2||A||_s.$$

Problem 9

Whats the application of count-min sketch? Please list at least three applications and describe in details. Each application should be described in detail more than one way.

Problem 10

Prove that the least storage required by any P-pass sorting algorithm for N elements is $\theta(\frac{N}{P})$.