Big Data Algorithms and Analysis Midterm Exam

Name: No.:

1. (Designer: Juntao Wang. You are required to answer this question) You are a project manage facing a tough task: finding a median profile among a extremely large quantity of profiles. Each profile is 1KB (2^{10}B), and all 64PB (2^{56}B) profiles are stored in a SSD server of capacity 1024PB (2^{60}B). But you only have an old mac with 4GB memory to connect the SSD server to finish the job. You already knew that the time for CPU to access a profile in the SSD server is 1000 times larger than accessing a profile in memory and that comparison of two profiles takes no time cost compared with the memory access time.

   (a) You recalled two fast finding median algorithms taught in class. One is the quick selection algorithm, which first subtly selects a pivot and divides data into two parts, then selects the part containing the median and recurses the procedure. The other is the one-pass algorithm that maintains part of data in memory and updates the memory taking a thought of random walks. Aiming to minimize the time cost in the worst case, which algorithm will you choose to solve the problem? Provide your worst-case running time estimation of both algorithms.

   (Hint: you can first figure out the constant coefficient of the linear time complexity of the first algorithm and analyze the time complexity of the one-pass algorithm, which haven't been discussed in class.)

   (b) Fortunately, you get to know that the profiles are independently drawn from an unknown identical distribution (i.i.d.). Could you design an even faster algorithm to compute a profile (expressed as a number) near the median profile with high probability. Provide your algorithm, analyze the running time, and present the approximation performance of your algorithm in the form of $Pr(|x - x^*| > \delta) < f(\delta)$, where x is your answer, x^* is the true median, δ is a variable or a number you define and f is a function you give.

   (Hint: think about algorithms not totally based on comparison.)

2. (Designer: Xiaofu Huang and Yun Guo) Analyze the time complex of the quick hull algorithm.The algorithm can be broken down to the following steps:

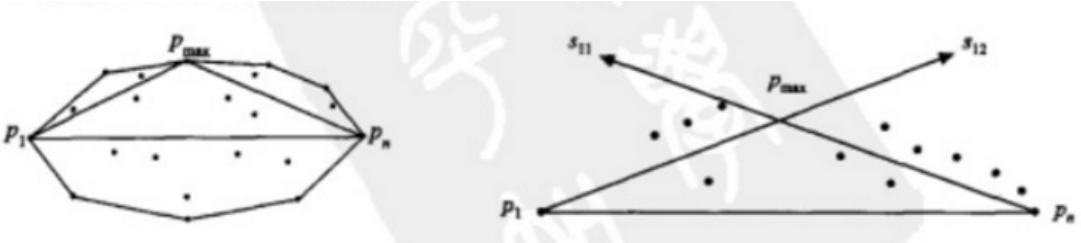

Figure 1: quick hall algorithm

- Find the points with minimum and maximum x coordinates, denoted as p_1 and p_n, those are bound to be part of the convex hull.
- Use the line formed by the two points (p_1, p_n) to divide the set in two subsets of points, which will be processed recursively.
- Determine the point (p_{max}), on one side of the line, with the maximum distance from the line. The two points found before along with this one form a triangle.

- The points lying inside of that triangle cannot be part of the convex hull and can therefore be ignored in the next steps.
- Repeat the previous two steps on the two lines formed by the triangle (not the initial line).
- Keep on doing so on until no more points are left, the recursion has come to an end and the points selected constitute the convex hull.

(a) whether the quick hull algorithm is output sensitive algorithm or not? Explain why. (An algorithm is output sensitive if its running time depends on the size of the output.)

(b) If h is the number of the points on the convex hall. For points independently drawn from a circular Gaussian distribution, it is known that $\mathbb{E}(h) = O(\sqrt{log(n)})$ for sufficiently large n, where n is the number of all points. What's the time complexity of the quick hall algorithm in this case?

3. (Designer: Xiaofu Huang and Yun Guo)

(a) Briefly describe the Kirkpatrick and Seidel algorithm for finding convex hall.

(b) Given n points in the plane and a partition principle below. Prove that the Kirkpatrick and Seidel algorithms complexity is $O(\min_{S_i} \sum_{i=1}^{k} |S_i| log(\frac{n}{|S_i|}))$.

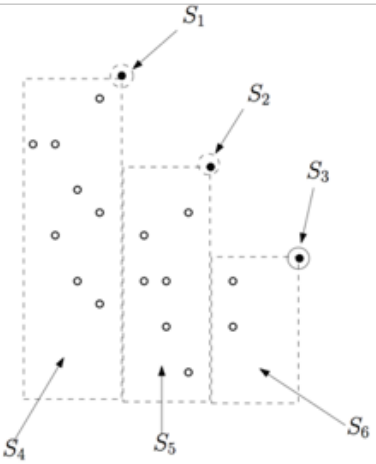

Figure 2: partition principle

The partition principle: we partition the input set S into groups ,..., where for each i:

- either S_i is a single point; or
- there is an axis-aligned box B_i such that its interior contains all of S_i and lies strictly below the"staircase" of S.

(Claim: Every set S_i has $O(\frac{n}{2^j})$ unpruned points remaining at the j-th recursion level of the KS algorithm.)

4. (Designer: Wenbo Li and Qipeng Zhang) Although computing shortest paths in graphs is a well-studied problem, algorithms such as Dijkstra and Floyd cant be adopted for nowadays real-world massive networks, considering the time and space complexity. Landmarks is a proper method for approximate solutions, which choose some vertices as landmarks, and calculated paths of acceptable lengths. Our problem, which is also called a landmarks-cover problem, is defined as follows: Given a graph $G = (V, E)$, select a minimum number of landmarks $L \subseteq V$, so that for all pairs of vertices (s, t), at least one landmark in L lies in the shortest path from s to t

(a) Prove that landmarks-cover problem is NP-hard. (Hint: reduce the vertex cover problem to this problem.)

(b) Design a (heuristic) method to generate the landmarks set, and analyze the potential fault.

5. (Designer: Jinli Zhong and Tong Yin) Proximity searching is the problem of, given a data set and a similarity criterion, finding the elements of the set that are close to a given query. This problem has a vast number of applications in data science such as text retrieve and DNA sequence comparison. A specific problem goes that given a set of points $p_1, ..., p_n$ in a metric space, a query q, which is also a point in the same metric space and a distance r, output all the points p_i such that $d(p_i, q) < r$, where $d(x, y)$ is the distance between two points x, y. Traditionally, we need to compute the distance of each point p_i and q, which takes n times of comparison operations. As the distance of two points in real application is really expensive to compute (think, for instance, in comparing two fingerprints), could you

   (a) come up with some ingenious algorithm to reduce the times of comparisons in average without harming the correctness. (Hint: you can precompute and save the distances of each pair of points in the data set, and when a query comes, you need not to compute these distances if you need to use them.)

   (b) calculate in expectation how many comparisons are reduced by your algorithm if we consider the matrix space $[0, 1]$, a set of two points $p_1 = \frac{1}{3}, p_2 = \frac{2}{3}$ and that q is uniformly drawn from $[0, 1]$ and r is uniformly drawn from $[0, \frac{2}{3}]$.

6. (Designer: Atta Ul Munim Zaki) Sketch is a useful technique in big data science. The idea of sketch is to compress data to keep the most desirable information and discard the rest. For example, in some applications, we may randomly drop most elements in a matrix in order to speed up the calculation, which is called random projection. However, let's deal with the simplest case. Consider the following problem:being a research manager of a mobile company in China, you are required to get the mean of some answer over aiming 269 million users. As it is quite impossible to know the willingness of all 269 million users, a suitable sample size should be chosen. So what sample size will you choose? Give you reason and analyze the accuracy of the mean evaluated by that sample size. (If you need more modest information that is able to access in practice in order to estimate the suitable sample size, you can list these information and assume that you can get them anywhere.)

7. (Designer: Wei Yi and Huang Bo) Kolmogorov complexity $H(s)$ of a string s is the minimum length of a Turing machine program to generate s. Assume the alphabet set of the Turing machine is $\{0, 1\}$.

   (a) Prove $\exists c \in \mathbb{N}$, $H(s) \leq |s| + c$ for any finite string s.

   (b) Prove $\exists c \in \mathbb{N}$, $H(< x, y >) \leq |x| + |y| + 2\log(|x|) + c$ for any a pair of finite strings x and y.

   (Hint: $H(< x, y >)$ is the minimum length of a Turing machine program to generate a string from which we can decode out x and y.)

8. (Designer: Haixing Huang and Zhen Meng) Vertex cover is a typical fixed parameter problem, it has a variation called positive integer weighted vertex cover, the description of the problem is as follows: Given a graph $G = (V, E)$, a positive integer k and a positive weight $w : V \to \mathbb{N}^+$ for each vertex, find a subset $C \subseteq V$ such that for all $(u, v) \in E$, at least one of u or v is contained in C and $\sum_{v \in C} w(v)$ is less than or equal to k. Find an algorithm to solve the problem above, and give its time complexity.

9. (Designer: Yuding Liang and Ze Chen) Order statistics $X_{r,n}$ is the random variable representing the r-th smallest value among n variables $X_1, ..., X_n$ randomly drawn from some joint distribution.

   (a) There are three correlated (random) variables Y_1, Y_2 and Y_3 defined as $Y_1(X) = 1 - 2X$, $Y_2(X) = 3X$ and $Y_3(X) = 1/6$ where $X \in [0, 1]$ is a random variable. Please represent the order statistics $Y_{r,3}$ with Y_1, Y_2, Y_3 for different value of X.

   (b) Suppose a sequence of random variables $X_1, ..., X_n$ are independently drawn from cdf F with support set $[0, 1]$ and let $F_{r,n}$ be the cdf of the order statistics $X_{r,n}$. Please show that

      i. $F_{r,n}(t) = F_{r+1,n}(t) + C_r^n(F(x))^r(1 - F(t))^{n-r}$
      ii. $F_{r,n}(t) = F_{r,n-1}(t) + C_{r-1}^{n-1}(F(x))^r(1 - F(t))^{n-r}$

   (c) $X_1, ..., X_{n+1}$ are independently drawn from the distribution with cdf F on support set $[0, 1]$. Let Y be the r-th smallest value of the first n variables and Z be the $(r + 1)$-th smallest value of the whole sequence. Please compute $Pr(Y \leq y, Z > y)$ for all $x \leq y$.

10. (Designer: Renjie Jin and Jie Li Liang) Order statistics $X_{r,n}$ is the random variable representing the r-th smallest value among n variables $X_1, ..., X_n$ randomly drawn from some joint distribution. Assume all X_i are independently drawn from the exponential distribution with pdf $f(x) = \lambda e^{-\lambda x}$.

    (a) Calculate the distribution of $X_{1,10}$.

    (b) Suppose that $X_{2,10} = 3$ after each variable is revealed, what is the distribution of $X_{3,10}$ now.

    (c) Assuming that the time to generate a variable from the exponent distribution is $O(1)$, design an algorithm to generate the k-th smallest number in time complexity $O(k)$.