Take-Home Final — *Applied Algorithms, T. II/2012–23*
(due: sat apr 8 @ 11:59pm)

Ground Rules:

- There are 3 problems. Do *any* 2 of them; if you do all 3, the worst 2 will be credited. Solve them by yourself—no collaboration.
- Typeset your answers and hand in a PDF file electronically to Canvas.
- You can look things up on the Internet and elsewhere; refrain from blindly copying solutions. If you do look something up, please cite your sources.
- For multi-part problems, if you can't solve a subproblem but need it subsequently, you can pretend you've solved it and invoke it in the following parts.

Problem 1. [100 points] *Triangle Count Estimation.* Let $G = (V, E)$ be an undirected graph with no loops and no parallel edges. A triplet $\{u, v, w\}$ is triangle in G if u, v, and w form a 3-vertex cycle. So, in this notation, $\{u, v, w\}$ and $\{v, w, u\}$ denote the same triangle. The goal here is to estimate the number of triangles in G without actually counting it. **Your Task:** Analyze the following method. An α-estimator algorithm proceeds as follows:

1. Pick an edge $e = \{u, v\}$ uniformly at random from E.
2. Pick a vertex w uniformly at random from $V \setminus \{u, v\}$.
3. Return $\alpha = 1$ if $\{u, v, w\}$ is a triangle[1] and $\alpha = 0$ otherwise.

(i) Prove that if α is the return value of the α-estimator algorithm (above), then

$$\mathbf{E}[\alpha] = \frac{3\tau}{m(n-2)},$$

where τ is the total number of triangles in G, $m = |E|$, and $n = |V|$.

(ii) We hope to use α to estimate τ, but it is unlikely that the return value of *one* α-estimator is good enough. A standard trick is to run multiple copies of the estimator. Suppose we run T such estimators. Let $\alpha_1, \alpha_2, \ldots, \alpha_T$ be their results. Show that if $\widehat{\alpha} = \frac{1}{T}(\alpha_1 + \alpha_2 + \cdots + \alpha_T)$ and $\widehat{\tau} = \frac{m(n-2)}{3} \cdot \widehat{\alpha}$, then $\mathbf{E}[\widehat{\tau}] = \tau$.

(iii) Let $\varepsilon \geq 0$ and $0 < \delta \leq 1/2$. Using this scheme, how large does T have to be in order for $\widehat{\tau}$ to satisfy $\mathbf{Pr}[|\widehat{\tau} - \tau| < \varepsilon] \geq 1 - \delta$? (*Hint:* T can be a function of τ. Use Chernoff-Hoeffding.)

Remarks: This algorithm can be implemented in the streaming setting, as it can be made such that we don't need to have the whole graph at any point in the computation. The algorithm presented here has been simplified for the purpose of this exam.

Problem 2. [100 points] *Closest Pair of Points in 2D.* Finding the closest pair of points in an input with n two-dimensional points has a well-known divide-and-conquer algorithm, which runs in $O(n \log n)$ time and can be made parallel without much brain damage. In class, we saw another algorithm that takes advantage of grid and hashing to solve the same problem in $O(n)$ expected time.

In this problem, we'll explore the linear-time algorithm a bit more.

(i) The linear-time algorithm can have a few "reboots" (i.e., when the grid size is adjusted and all the points from the start have to be added again). Assuming that the points have been randomly shuffled, what's the expected number of "reboots" if the algorithm is run with n points? Show your work.

Extra-credit: Show this to be true with high probability.

[1] It is easy to check this because we know e is an edge for sure, so it remains to check if w is incident on both u and v.

(ii) In practice, how does the $O(n \log n)$-time divide-and-conquer algorithm perform compared with the $O(n)$-time algorithm in terms of time and space usage? Design an experiment to compare the algorithms and discuss your findings.

Problem 3. [100 points] *Parallel Sublinear-Work Approximate Median.* This problem has two main parts. First, we're going to develop a parallel algorithm to find the median in an array. Then, we're going to look at developing a sampling scheme that will help us with a better work bound but will give us an approximate median.

Given an array A of numbers, define the rank of u with respect to A, denoted by $\text{rank}_A(u)$, to be the number of elements in A that are less than or equal to A. For the purpose of this problem, the *median* of A is the smallest number $x \in A$ such that $\text{rank}(x) \geq \texttt{len}(A)/2$. We'll assume A has distinct entries. Simply put, the median is the element $\texttt{sorted}(A)[n//2]$, where $n = \texttt{len}(A)$.

(i) Describe an algorithm that finds the median in expected $O(n)$ work and $O(\log^c n)$ span with high probability (for some $c \geq 1$). Prove that the algorithm meets the bounds. (*Hint:* Randomized quickselect.)

(ii) Sampling with replacement (see the Internet for more explanation) is when after a sample x is drawn, that x is returned back to the pool and continues to be available for further sampling. This means an item x can be drawn multiple times. Explain how you can sample T elements with replacement from a (random-access) array A of length n in $O(T)$ work and $O(\log n)$ span, assuming each `randint(low, high)` takes $O(1)$ work and span.

(iii) Let X be an array of n *distinct* numbers. An ε-approximate-median of X is an element $y \in X$ such that $\frac{n}{2} - \varepsilon n < \text{rank}_X(y) < \frac{n}{2} + \varepsilon n$.

Your Task: You'll derive an ε-approximate-median that runs in $o(n)$ work (little-O of n) and $O(\text{polylog}(n))$ span. Here's an outline of how you want to proceed:

1. For a parameter $T > 0$ (that you'll set), use the previous part to sample T items with replacement from X. Call the resulting array A.
2. Run the algorithm in part (i) to find the median on A.

Because you will use $T = o(n)$, the work required will be $O(T)$. The main line of argument you'll need to show is that for your setting of T, good things will happen—i.e., the answer your algorithm returns is an ε-approximate-median with high-probability.

To analyze this, you may find the following setup and observation useful: Fix $\varepsilon > 0$. Let

$$S = \left\{x \in A \mid \text{rank}_X(x) \leq \tfrac{n}{2} - \varepsilon n\right\} \qquad L = \left\{x \in A \mid \text{rank}_X(x) \geq \tfrac{n}{2} + \varepsilon n\right\},$$

as determined by their ranks in X (not in A). Now convince yourself that if $|S| < T/2$ and $|L| < T/2$, then the median of A is an ε-approximate-median of X.

In your write-up, be sure to indicate the value of T you end up with, prove that this setting of T gives you the desired high probability bound, and analyze the total cost (work and span). You don't need to pinpoint the constants in T; it's okay to say use $T = \Theta(\cdots)$ or $T = \Omega(\ldots)$.

(*Hint:* Be inspired by your A2.)