Algorithm Design and Analysis (Fall 2022)

Assignment 1

Deadline: Oct 14, 2022

1. (20 Points) Prove the following generalization of the master theorem. Given constants $a \geq 1, b > 1, d \geq 0$, and $w \geq 0$, if $T(n) = 1$ for $n < b$ and $T(n) = aT(n/b) + n^d \log^w n$, we have

$$T(n) = \begin{cases} O(n^d \log^w n) & \text{if } a < b^d \\ O(n^{\log_b a}) & \text{if } a > b^d \\ O(n^d \log^{w+1} n) & \text{if } a = b^d \end{cases}.$$

2. (15 points) A k-way merge operation. Suppose you have k sorted arrays, each with n elements, and you want to combine them into a single sorted array of kn elements. Design an efficient algorithm using divide-and-conquer (and give its time complexity).

3. (15 points) Given two sorted lists $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$. Design a divide and conquer algorithm to find the median of the union of the two lists. Your algorithm should have running time better than $O(n)$.

4. (25 points) An integer triple (x, y, z) forms a *good order* if $y - x = z - y$. Given an integer n, we call a permutation of $\{1, \ldots, n\}$ (denoted by $X = \{x_1, x_2, \ldots, x_n\}$) is *out-of-order* if it has the following property: for every $i < j < k$, the triple (x_i, x_j, x_k) does not form a *good order* (i.e., $x_j - x_i \neq x_k - x_j$).

   (a) (3 points) Prove that an integer triple (x, y, z) forms a *good order* only if x and z are both even or both odd.

   (b) (5 points) Write down a permutation of $\{1, \ldots, 4\}$, and quickly prove it is *out-of-order* by the claim above.

   (c) (3 points) Prove that an integer triple (x, y, z) where x, y, z are all odd forms a good order only if $((x+1)/2, (y+1)/2, (z+1)/2)$ forms a *good order*.

   (d) (14 points) Design a divide and conquer algorithm runs in $O(n \log n)$ time to output an *out-of-order* permutation of $\{1, \ldots, n\}$. (The running time can be improved to $O(n)$.)

5. (25 points) In the *cake cutting* problem, you are going to allocate a piece of birthday cake to a group of n children. Different parts of the cake may have different toppings: some parts may be covered by chocolate, some parts may be covered by strawberry, some parts may be covered by pineapple, and so on. Different children may have different preferences over different parts of the cake. The objective is to allocate the cake to the n children *fairly* such that each child believes he receives the average value based on his preference. This is formally modelled as follows.

   The cake is modelled by the 1-dimensional interval $[0, 1]$. Each child i has a *value density function* $f_i : [0, 1] \to \mathbb{R}_{\geq 0}$ representing i's preference over the cake $[0, 1]$. Given an interval $I \subseteq [0, 1]$, child i's *value* on I is given by the Riemann integral $\int_I f_i(x)dx$. An *allocation* is a collection of n intervals $(I_1, \ldots, I_n)$ such that every pair of intervals I_i and I_j can only intersect at the endpoints, where I_i is the interval allocated to child i. An allocation $(I_1, \ldots, I_n)$ is *proportional* if $\int_{I_i} f_i(x)dx \geq \frac{1}{n}\int_0^1 f_i(x)dx$ for each child i, i.e., each child i thinks he receives the average value based on his value density function. In this question, you are going to analyze a classical cake cutting algorithm, *the Even-Paz algorithm*, which is described below.

   For each child i, a half-half point $x_i \in [0, 1]$ is computed such that the ratio of the values for the two pieces $[0, x_i]$ and $[x_i, 1]$ is $\lfloor \frac{n}{2} \rfloor : \lceil \frac{n}{2} \rceil$. That is

$$\int_0^{x_i} f_i(x)dx = \frac{1}{n}\left\lfloor \frac{n}{2} \right\rfloor \cdot \int_0^1 f_i(x)dx \quad \text{and} \quad \int_{x_i}^1 f_i(x)dx = \frac{1}{n}\left\lceil \frac{n}{2} \right\rceil \cdot \int_0^1 f_i(x)dx.$$

   Next, we find the child i^* such that exactly $\lfloor \frac{n}{2} \rfloor$ children's half-half points are less than or equal to child i^*'s half-half point x_{i^*}. Let S be the set of those $\lfloor \frac{n}{2} \rfloor$ children, and $\bar{S}$ be the set of the remaining children. The cake is then cut to two parts $[0, x_{i^*}]$ and $[x_{i^*}, 1]$. An allocation of $[0, x_{i^*}]$ to the children in S and an allocation of $[x_{i^*}, 1]$ to the children in $\bar{S}$ are then computed *recursively*.

   (a) (8 points) Write down the Even-Paz algorithm in pseudo-codes.

   (b) (7 points) Suppose the half-half point x_i for each child i can be computed in $O(1)$ time. Analyze the time complexity of the Even-Paz algorithm.

   (c) (10 points) Prove that the allocation returned by the Even-Paz algorithm is proportional.

6. How long does it take you to finish the assignment (including thinking and discussion)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Please write down their names here.