CS 598TMC, Fall 2022
Timothy Chan

Homework 1 (due Sep 22 Thursday 5pm)

Instructions: You may work individually or in groups of at most 3; submit one set of solutions per group. Always acknowledge discussions you have with other people and any sources you have used (although most homework problems should be doable without using outside sources). In any case, *solutions must be written in your own words.*

1. [*25 pts*] Given n integers $a_1, \ldots, a_n, b_1, \ldots, b_n$, define the function

$$F(x) = \frac{1}{a_1 x + b_1} + \frac{1}{a_2 x + b_2} + \cdots + \frac{1}{a_n x + b_n}.$$

Then $F(x)$ is a *rational function*, i.e., a ratio of two polynomials (of degree at most n). Describe an $O(n \log^2 n)$-time algorithm to compute the coefficients of these polynomials. In other words, compute $c_0, \ldots, c_{n-1}, d_0, \ldots, d_n$ such that

$$F(x) = \frac{c_{n-1}x^{n-1} + \cdots + c_0}{d_n x^n + d_{n-1}x^{n-1} + \cdots + d_0}.$$

You may assume that arithmetic operations on numbers take constant time (even though the coefficients could be very large).

(*Hint*: divide-and-conquer. Use convolution/FFT as subroutine.)

2. [*25 pts*] Consider the following variant of the convolution problem, called the *plus-max convolution problem*: Given sequences $\langle a_0, \ldots, a_{n-1} \rangle$ and $\langle b_0, \ldots, b_{n-1} \rangle$, compute the sequence $\langle c_0, \ldots, c_{2n-2} \rangle$ where

$$c_i = \sum_{j=0}^{i} \max\{a_j, b_{i-j}\}.$$

Assume that all the input numbers are distinct. Let $S = \{a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}\}$ be the set of all $2n$ input numbers. Sort S, and divide the sorted list into $2n/\Delta$ sublists each of length Δ (ignoring floors and ceilings), for some parameter Δ. Call these sublists *buckets*.

(a) Define

$$c'_i = \sum_{j \in \{0,\ldots,i\}:\ a_j \text{ and } b_{i-j} \text{ are in different buckets}} \max\{a_j, b_{i-j}\}.$$

First describe an $O((n/\Delta) \cdot n \log n)$-time algorithm to compute all of these values c'_i.

(b) Next define

$$c''_i = \sum_{j \in \{0,\ldots,i\}:\ a_j \text{ and } b_{i-j} \text{ are in the same bucket}} \max\{a_j, b_{i-j}\}.$$

Describe an $O(\Delta n)$-time algorithm to compute all of these values c''_i.

(c) Finally describe an $\widetilde{O}(n^{1.5})$-time algorithm to solve the plus-max convolution problem. (As usual, the $\widetilde{O}$ notation hides logarithmic factors.)

3. [*25 pts*] Consider the following variant of subset sum: We are given a set S of n positive integers, a target number T, and numbers k and ℓ (with $k \leq \ell$). We want to find a multiset $R \subseteq S$ (allowing duplicates) that sums to T, such that the number of distinct elements in R is at most k and the number of elements in R (including duplicates) is at most ℓ.

   Describe an algorithm with running time $\widetilde{O}(\ell^2 T)$, or better still, $\widetilde{O}(\ell T)$.

   (For example, for $S = \{5, 7, 11, 20\}$, $T = 38$, $k = 2$, and $\ell = 4$, a solution is $\{5, 11, 11, 11\}$.)

4. [*25 pts*] We are given a collection of n strings $s^{(1)}, \ldots, s^{(n)} \in (\Sigma \cup \{\texttt{?}\})^*$ with the "don't care" symbol `?`. All strings are of length ℓ.

   We want to determine whether there exist two strings $s^{(i)}$ and $s^{(j)}$ ($i \neq j$) that *match*. Here, two strings $a_1 \cdots a_\ell$ and $b_1 \cdots b_\ell$ in $(\Sigma \cup \{\texttt{?}\})^*$ are said to match iff for all $k \in \{1, \ldots, \ell\}$, we have $a_k = b_k$ or $a_k = \texttt{?}$ or $b_k = \texttt{?}$.

   The trivial algorithm runs in $O(\ell n^2)$ time. Describe a faster algorithm.

   [Hint: use the idea from Clifford and Clifford's algorithm, but replace convolution with matrix multiplication. . .]