Algorithm Design and Analysis

Assignment 5

1. Consider the following 3-PARTITION problem. Given integers $a_1, ..., a_n$, we want to determine whether it is possible to partition of $\{1, ..., n\}$ into three disjoint subsets I, J, K such that

$$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k = \frac{1}{3} \sum_{i=1}^{n} a_i.$$

   For example, for input $(1, 2, 3, 4, 4, 5, 8)$ the answer is yes, because there is the partition $(1, 8)$, $(4, 5)$, $(2, 3, 4)$. On the other hand, for input $(2, 2, 3, 5)$ the answer is no. Devise and analyze a dynamic programming algorithm for 3-PARTITION that runs in time polynomial in n and in $\sum_i a_i$.

2. Let $X[1..n]$ be a reference DNA sequence. Let S be a set of m exon candidates of X, where each exon candidate is represented by a triple (i, j, w), which means that the strength or probability for fragment $X[i..j]$ being an exon is w. Notice that many triples in S are false exons, and true exons do not overlap. Show how to use dynamic programming to find a maximum-weight subset of S in which all exon candidates are non-overlapping. The time complexity should be linear in terms of n and m.

3. Consider the following game. A "dealer" produces a sequence $s_1, \ldots, s_n$ of "cards" facing up, where each card s_i has a value v_i. Then two players take turns picking a card from the sequence, but can only pick the first or the last card of the (remaining) sequence. The goal is to collect cards of largest total value. Assume n is even.

   (a) Show a sequence of cards such that it is not optimal for the first player to start by picking up the available card of larger value. That is, the natural *greedy* stratgy is suboptimal.

   (b) Give an $O(n^2)$ algorithm to compute an optimal strategy for the first player. Given the initial sequence, your algorithm should precompute in $O(n^2)$ time some information, and then the first player should be able to make each move optimally in $O(1)$ time by looking up the precomputed information.

4. Assume points $v_1, v_2, \ldots, v_n$ form a convex polygon in $\mathbb{R}^2$. Let $d(i,j)$ be the Euclidean distance between v_i and v_j if $i \le j$ and $d(i,j) = -\infty$ if $i > j$. For every $r \ge 0$, we use $d^{(r)}(i,j)$ to denote the length of the the *longest paths* from v_i to v_j using *at most* r edges. Therefore, $d(i,j) = d^{(1)}(i,j)$.

   (a) Let $s, t \ge 0$ be any two any integers satisfying $r = s + t$. For every $i \le j$, prove that $d^{(r)}(i,j) = \max_{i \le k \le j} \left\{ d^{(s)}(i,k) + d^{(t)}(k,j) \right\}$.

   (b) Prove that the distance $d(\cdot,\cdot)$ satisfies the *inverse Quadrangle Inequality* (iQI):

   $$\forall i \le i' \le j \le j' : d(i,j) + d(i',j') \ge d(i',j) + d(i,j').$$

   (c) Prove that for any integer $r \ge 0$, $d^{(r)}(\cdot,\cdot)$ satisfies iQI as well.

   (d) If we let $K^{(r)}(i,j)$ denote $\max \left\{ k \mid i \le k \le j \text{ and } d^{(r)}(i,j) = d^{(s)}(i,k) + d^{(t)}(k,j) \right\}$, prove that

   $$K^{(r)}(i,j) \le K^{(r)}(i,j+1) \le K^{(r)}(i+1,j+1), \quad \text{for} \quad i \le j.$$

   (e) Give an algorithm to compute $d^{(r)}(i,j)$ for all $1 \le i < j \le n$ in $O(\log r \cdot n^2)$ time [1].

5. How long does it take you to finish the assignment (include thinking and discussing)? Give a score (1,2,3,4,5) to the difficulty.

[1] That is, your algorithm needs to compute all the $\binom{n}{2}$ values within $O(\log r \cdot n^2)$ time.