Algorithm Design and Analysis

Assignment 4

Deadline: May 9, 2022

1. (30 points) Given a sequence of integers $a_1, a_2, ..., a_n$, a lower bound and and upper bound $1 \leq L \leq R \leq n$. An (L, R)-step subsequence is a subsequence $a_{i_1}, a_{i_2}, ..., a_{i_\ell}$, such that $\forall 1 \leq j \leq \ell - 1$, $L \leq i_{j+1} - i_j \leq R$. The revenue of the subsequence is $\sum_{j=1}^{\ell} a_{i_j}$. Design a DP algorithm to output the maximum revenue we can get from a (L, R)-step subsequence.

   (a) (10 points) Suppose $L = R = 1$. Design a DP algorithm in $O(n)$ to find the maximum $(1, 1)$-step subsequence.

   (b) (10 points) Design a DP algorithm in $O(n^2)$ to find the maximum (L, R)-step subsequence for any L and R.

   (c) (10 points) Design a DP algorithm in $O(n)$ to find the maximum (L, R)-step subsequence for any L and R. (Tips: Refer to the "Priority Queue" technique of the k-largest number problem in the lecture.)

2. (25 points) **Optimal Indexing for A Dictionary:** Consider a dictionary with n different words $a_1, a_2, ..., a_n$ sorted by the alphabetical order. We have already known the number of search times of each word a_i, which is represented by w_i. Suppose that the dictionary stores all words in a binary search tree T, i.e., each node's word is alphabetically larger than the words stored in its left subtree and smaller than the words stored in its right subtree. Then, to look up a word in the dictionary, we have to do $\ell_i(T)$ comparisons on the binary search tree, where $\ell_i(T)$ is exactly the level of the node that stores a_i (root has level 1). We evaluate the search tree by the total number of comparisons for searching the n words, i.e., $\sum_{i=1}^{n} w_i \ell_i(T)$. Design a DP algorithm to find the best binary search tree for the n words to minimize the total number of comparisons.

3. (25 points) A *palindrome* is a nonempty string over some alphabet that reads the same forward and backward. Examples of palindromes are all strings of length 1, `civic`, `racecar`, and `aibohphobia` (fear of palindromes).

   Give an efficient algorithm to find the longest palindrome that is a subsequence of a given input string. For example, given the input `character`, your algorithm should return `carac`. What is the running time of your algorithm?

4. (25 points) Let G be a tree with n vertices. In this problem, we assume that it takes $O(1)$ time to store and multiply two integers.
   (a) (20 points) Design an $O(n)$ time algorithm to count the number of independent sets in G. Prove the correctness of your algorithm and analyze its time complexity.
   (b) (Bonus 5 points) Design an efficient algorithm to count the number of *maximum* independent sets in G. Prove the correctness of your algorithm and analyze its time complexity.
5. How long does it take you to finish the assignment (including thinking and discussion)?

   Give a rating (1,2,3,4,5) to the difficulty (the higher the more difficult) for each problem.

   Do you have any collaborators? Please write down their names here.