CMPS 6610 Algorithms – Fall 18

10/8/18

5. Homework

Due **10/15/18** at the beginning of class

Justify all your answers.

1. **Multi-Min (8 points)**
   Consider the following task: Given an unsorted array of n numbers, find the k smallest numbers and output them in sorted order.
   Describe three inherently different algorithms that solve this problem. Analyze their runtimes in terms of n and k (so you should have n and k in the big-Oh runtime). Try to find the fastest possible algorithm. Which of your algorithms is the fastest?

2. **Halloween (10 points)**

   It is October and you are preparing for Halloween. You have a fixed budget of B dollars to spend, and you found three different types of candy that you are considering to buy: Caramel apples, chocolate bars, and gummy bears. Of course your goal is to maximize the happiness of the children you give the candy to, but at the same time you have to stay within your budget. The three different types of candy have different costs: $c_{apples} > c_{chocolate} > c_{gummy}$. But you know that while some of the children like gummy bears the best (which is great because it is the cheapest), some children prefer the caramel apples and some prefer the chocolate bars. In fact, you know the children that come by your door at Halloween pretty well, and you know all of their candy preferences. For the i-th child, let $H_{apples}[i]$, $H_{chocolate}[i]$, $H_{gummy}[i]$ be the child's happiness for the respective candy type. These are all positive integer values.

   Suppose there are n children. Your goal is to buy one piece of candy for each child such that the total happiness of all children is maximized, while the total cost of the candy is within your budget B.

   (a) (2 points) Suppose there are $n = 4$ children, and $c_{apples} = \$5$, $c_{chocolate} = \$2$, $c_{gummy} = \$1$. And the happiness values are given in the following arrays:
   $H_{apples} = [4, 1, 2, 4]$, $H_{chocolate} = [1, 3, 4, 2]$, $H_{gummy} = [2, 1, 3, 3]$
   What selection of candies maximizes total happiness if you can spend at most $B = \$9$? What is the solution for $B = \$8$?

   (b) (2 points) Let $h(i, b)$ be the maximum total happiness for the first i children with total cost less than or equal to b. Provide a recursive formula for $h(i, b)$.

   (c) (3 points) Give pseudocode for a dynamic programing algorithm to compute $h(n, B)$. What are the runtime and the space complexity of your algorithm?

   (d) (3 points) Give pseudocode for tracing back in the DP table to compute an optimal candy selection. What is the runtime of your algorithm?

FLIP OVER TO NEXT PAGE $\Longrightarrow$

3. **Intervals (10 points)**
Let $A[1..n]$ be an array of n integers (which can be positive, negative, or zero). An *interval* with start-point i and end-point j, $i \leq j$, consists of the numbers $A[i], \ldots, A[j]$ and the *weight* of this interval is the sum $A[i] + \ldots + A[j]$.

The problem is: Find the interval in A with maximum weight.

(a) (2 points) Describe an algorithm for this problem that is based on the following idea: Try out all combinations of i, j with $1 \leq i < j \leq n$. What is the runtime of this algorithm?

(b) Describe a dynamic programming algorithm for this problem. Proceed in the following steps:

i. (2 points) Develop a recursive formula for the following entity: $S(j) =$ maximum of the weights of all intervals with end-point j.

ii. (2 point) Describe an algorithm that computes all $S(j)$ in a dynamic programming fashion based on the recursive formula, and afterwards determines the end-point j^* of an optimal interval. You can describe your algorithm in words.

iii. (2 points) Given the end-point j^* describe how to find the start-point i^* of an optimal interval by backtracking.

iv. (2 point) What are the runtime and the space complexity of the overall algorithm?

Practice Question

This question is just for practice and not for credit.

Deterministic Select
The deterministic select algorithm that we covered in class splits the elements into groups of 5.

(a) Argue that when using groups of 3 the runtime analysis to show linear runtime fails.

(b) Prove that the runtime of the algorithm is $O(n)$ when splitting the elements into groups of 7.