Prof. Salma Elmalaki
Fall 2022

Homework 3
Practicing Chapter 5 (DC) and 6 (DP)

1. *(15 Points)* Asymptotic running time of divide-and-conquer algorithms:

   (a) *(10 Points)* **Master Theorem.** If $T(n) = aT(n/b) + O(n^d)$ for some constants $a > 0, b > 1, d \geq 0$, then prove that it is:

   $$T(n) = \begin{cases} O(n^d) & \text{if } d > log_b a \\ O(n^d logn) & \text{if } d = log_b a \\ O(n^{log_b a}) & \text{if } d < log_b a \end{cases}$$

   (b) *(5 Points)* Then show that the divide-and-conquer algorithms we did in class Mergesort, Binary Search, Matrix multiplication (classic) and Fast Matrix Multiplication (Strassen) are all special cases of the Master Theorem. In particular, for each of these problems: (i) give the parameters a, b, d for each algorithm and (ii) write down the recursive and closed form equation for $T(n)$.

2. *(20 Points)* **Chapter 5: More recursions:** Solve the following recurrence relations and give a O bound for each of them:

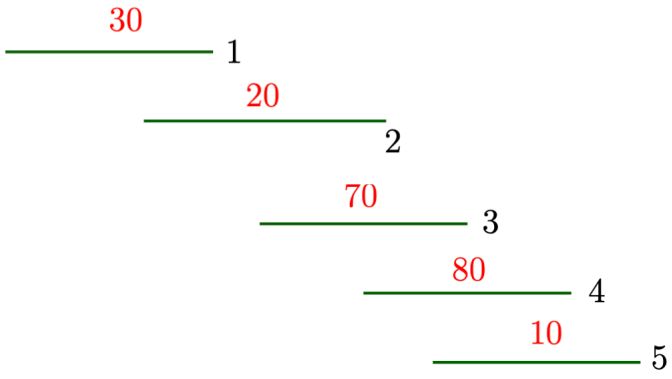
   (a) *(5 Points)* $T(n) = 2T(n/3) + 1$

   (b) *(5 Points)* $T(n) = 5T(n/4) + n$

   (c) *(5 Points)* $T(n) = 9T(n/3) + n^2$

   (d) *(5 Points)* $T(n) = 2T(n-1) + 1$

3. *(5 points)* **Chapter 6: Dynamic Programming: WIS via DAG** Consider the intervals below, each interval i has start time and a finish time and a weight v_i indicated on top of the interval i in the figure.



(a) *(2.5 points)* First create a DAG for these intervals as follows: Each node represents one interval. For each interval i add edge $(i, p(i))$ of the length/weight of v_i. You will need two extra nodes One dummy sink node 0 for interval 0 (Remember in solving WIS we say that p(1) = 0, so we need to have a dummy 0 interval as a sink node), plus another dummy source node s connected with the last interval (node) 5. Add an edge from s to 5 of length 0. For each interval i add edge $(i, i-1)$ of length 0. Second what is the running time of creating this DAG, assuming that the intervals were initially sorted.
nlogn

(b) *(2.5 points)* Now given a general interval problem instance I let and let $G(I)$ denote a DAG constructed similar to the one you did in the previous problem. You need to find the optimal solution for the weighted interval scheduling. Can you solve this problem by running Dijkstra? Why?

4. *(30 Points)* **Chapter 5: Variations of Knapsack Problem: Multidimensional Knapsack.** Consider each item $i = 1, ...n$ has a weight w_i, a volume z_i and a value v_i. You still want to choose a subset of the n items so as to maximize total value. However, you now need to meet two constraints: (1) the total weight of the selected items should not exceed a given W and (2) the total volume of the selected items should not exceed a given Z.

   (a) *(10 Points)* Design a DP algorithm that finds the optimal solution to this problem.

   (b) *(5 Points)* Prove that your algorithm finds indeed the optimal solution.

   (c) *(5 Points)* Analyze the running time.

   (d) *(10 Points)* Consider the example problem from class: there are $n = 5$ items with values $\{v_1 = 1, v_2 = 6, v_3 = 18, v_4 = 22, v_5 = 28\}$ and weights $\{w_1 = 1, w_2 = 2, w_3 = 5, w_4 = 6, w_5 = 7\}$; the constraint on the total weight is $W = 11$. In addition, consider that the items have volumes $\{z_1 = 1, z_2 = 10, z_3 = 3, z_4 = 2, z_5 = 2\}$ and that the total allowed volume is $Z = 15$. Apply your algorithm to solve this example.

5. *(30 Points)* **Chapter 5: Variations of Knapsack Problem: Knapsack with Repetition.**

   (a) *(10 Points)* Given n types of items (with weights $w_1, ..., w_n$ and values $v_1, ..., v_n$, respectively), describe a DP algorithm that finds the most valuable set of items, subject to a total weight constraint W. You are allowed to use repetition, i.e., to use items of the same type multiple times. What is the asymptotic running time of your solution? Apply your algorithm to the example with the following $n = 4$ items and capacity $W = 10$:

Item i	1	2	3	4
Weight w_i	6	3	4	2
Value v_i	30	14	16	9

   (b) *(10 Points)* For the above example, draw the graph with nodes the DP subproblems and edges the dependencies between them. Is it a DAG? Can you re-state this particular variant of the KP problem as finding a path on this graph?

   (c) *(10 Points)* Implement one of your DP algorithms, in `Python`, and run it on the example above. You should submit your code - the grader should be able to run your code with the input text file (format and example provided on Canvas - see `KP all.ZIP` and `README` therein) and test that it gives the correct output!

(a) *(10 Points)* Given n types of items (with weights $w_1, ..., w_n$ and values $v_1, ..., v_n$, respectively), describe a DP algorithm that finds the most valuable set of items, subject to a total weight constraint W. You are allowed to use repetition, i.e., to use items of the same type multiple times. What is the asymptotic running time of your solution? Apply your algorithm to the example with the following $n = 4$ items and capacity $W = 10$:

Item i	1	2	3	4
Weight w_i	6	3	4	2
Value v_i	30	14	16	9

(b) *(10 Points)* For the above example, draw the graph with nodes the DP subproblems and edges the dependencies between them. Is it a DAG? Can you re-state this particular variant of the KP problem as finding a path on this graph?

(c) *(10 Points)* Implement one of your DP algorithms, in `Python`, and run it on the example above. You should submit your code - the grader should be able to run your code with the input text file (format and example provided on Canvas - see `KP all.ZIP` and `README` therein) and test that it gives the correct output!