CME 2204 ALGORITHM ANALYSIS

Dynamic Programming Assignment

Report

Completed Tasks:

In this project, I have completed all the tasks requested from us. The tasks I have completed are as follows: I read the yearly player demands and player salary information in the files given to us and assigned them to the arrays in my program. Using the Dynamic Programming technique, I coded a function that takes n as the year, c as the coach cost, p as the yearly number of players that can be produced, and the information in the arrays as inputs. By using this function, I have ensured that the minimum cost is found under the desired conditions. I also showed how many players we need to produce at each step by using the traceback array in the code.

Uncompleted Tasks:

I have completed all the tasks requested from us in the assignment. By testing and debugging my program, I fixed my errors.

Run-Time Complexity:

When I examine the run-time complexity of the function I created by using a dynamic programming approach, I observed that the structure of the function consists of 3 nested loops and if else blocks. To calculate the runtime complexity, I needed to calculate how many times these 3 loops are repeated in total. To show how I do the calculations, I will show the calculations by writing the psuedo code version of the function below

for i to number of years ⟶ O(n+1), where n is the number of years

for j to total demand ⟶ O((n+1) * (total demand + 1))

for k to the number of possibilities that may occur in the previous step ⟶ O((n+1)*(total demand+1) *(totaldemand+1))

(it can go up to the maximum "total demand" value.)

if else blocks where calculations are made ⟶ O(c)

➔ O((n+1) + (n+1*total demand+1) + ((n+1)*(total demand+1) *(totaldemand+1)) + c)

Run Time Complexity = O(n * total demand * total demand)

In the runtime complexity calculation, in the outermost loop, all the years are iterated sequentially. In the loop inside it, as many as the number of extra players that can be produced during that year are iterated. In the innermost loop, all the possibilities of the number of players that can be taken from the previous year are iterated, this probability can be as much as the total demand. As a result, the run time complexity is equal to multiplying the square of the total demand by the number of years.

Space Complexity:

This function, written using a Dynamic Programming approach, uses two extra matrices to perform operations. If I calculate the size of these matrices and find the space complexity of the function. The number of rows of matrices is equal to one more than the number of years, and the number of columns is equal to one more than the total demand. These matrices are the same size and use an equal amount of space. As a result, if we calculate the space complexity of the function:

➔ O(2 * (n + 1) * (total demand + 1))

Space Complexity = O (n * total demand)