DOKUZ EYLUL UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

CME 2204 ALGORITHM ANALYSIS
ASSIGNMENT – II
REPORT

Dynamic Programming and Greedy Approach

by

S.Ayberk Kılıçaslan - 2017510053

Lecturers

Dr. Lec. Zerrin IŞIK

Res.Asst. Ali CÜVİTOĞLU

Res.Asst. Ezgi Demir

IZMIR

28.05.2020

INTRODUCTION

In this assignment, we supposed to implement dynamic programming and greedy approach to solve the given problem and compare the results in order to calculating time complexity and space complexity.

EXPLANATION OF ALGORITHMS

Dynamic Programming

Dynamic programming approach is similar to divide and conquer in breaking down the problem into smaller and yet smaller possible sub-problems. But unlike, divide and conquer, these sub-problems are not solved independently. Rather, results of these smaller sub-problems are remembered and used for similar or overlapping sub-problems.

Dynamic programming is used where we have problems, which can be divided into similar sub-problems, so that their results can be re-used. Mostly, these algorithms are used for optimization. Before solving the in-hand sub-problem, dynamic algorithm will try to examine the results of the previously solved sub-problems.

Greedy Approach

Greedy is an algorithmic paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit. So, the problems where choosing locally optimal also leads to global solution are best fit for Greedy.

For example consider the Fractional Knapsack Problem. The local optimal strategy is to choose the item that has maximum value vs weight ratio. This strategy also leads to global optimal solution because we allowed to take fractions of an item.

Part 1 Dynamic Programming –

I could not make this part correctly. You will see my results in the next calculations part so I will not explain this part. These calculations made by using your explanation in the lab session about this homework.

Some example results of this part :

TEST-1

P = 6; D=6; X= 5; T = 2; B=100; C = 6;

DP RESULTS-PROFIT: 62

TEST-2

P = 7; D=5; X= 20; T = 1; B=50; C = 4;

DP RESULTS-PROFIT: 212

TEST-4

P = 5; D=5; X= 30; T = 3; B=150; C = 4;

DP RESULTS-PROFIT: 348

Part 1 Greedy Approach –

In this part, we get from user a desired month and pay value for interns. We must calculate the minimum cost for each month so hold cars in the garage is not makes sense at all. Because holding car in the garage is way expensive then hiring an intern to greedy approach. That is why, I take demand and check for our capability to make a new car and calculate the minimum cost by hiring internships.

Some example results of this part :

TEST-1

P = 6; D=6; X= 5; T = 2; B=100; C = 6;

GREEDY RESULTS-COST: 36

TEST-2

P = 7; D=5; X= 20; T = 1; B=50; C = 4;

GREEDY RESULTS-COST: 70

TEST-4

P = 5; D=5; X= 30; T = 3; B=150; C = 4;

GREEDY RESULTS-PROFIT: 265

PART 2 DYNAMIC PROGRAMMING

I used 2D array to hold each month's profit dynamically because of to not lose any steps while calculating with different values. But there are some changes in my code about calculation of taxes. Now I will mention about my calculation and tell the difference between my and yours tax calculation for profit.

When I am looking for another company for next month, I am holding all money which comes from previous month and calculating the profit to tax if company is different. Because of, I am getting more money to calculate, I will earn more money or less money than your calculation result. Dynamic programming part works exactly perfect for each scenario, but my tax calculation is a bit different. You will see the difference of results, but I need to teach you how I calculate the profit. I hope you will understand my calculations and you will see my dynamic code is working very well.

Here are the example results,

Some example results of this part :

TEST-1

P = 6; D=6; X= 5; T = 2; B=100; C = 6;

DP RESULTS-PROFIT: 4644

TEST-2

P = 7; D=5; X= 20; T = 1; B=50; C = 4;

DP RESULTS-PROFIT: 19441

TEST-4

P = 5; D=5; X= 30; T = 3; B=150; C = 4;

DP RESULTS-PROFIT: 171222

PART 2 GREEDY APPROACH

Here this part, I used the same tax and profit calculations which I mentioned previous part. But this part is greedy, so I just calculated the best solution for each statement month by month. I get the best value for each month then I will calculate the tax and calculate the next month's profit. There is no 2D array in this solution. I used so many different temp values(double) and I hold every specified value in these temp values to assign them to themselves every step. Finally, I printed to the screen total cost after every loop iteration is done.

As a calculation, I hold the half of the first month's earning and switch them to the rest of the months. But each month that half earning swapped for that month's demand by multiplying with our B value. Total cost variable holds the maximum money which we get every step. Total cost is made by combining half earning and every month earning with the investment rate.

Some example results of this part :

TEST-1

P = 6; D=6; X= 5; T = 2; B=100; C = 6;

GREEDY RESULTS-PROFIT: 4443

TEST-2

P = 7; D=5; X= 20; T = 1; B=50; C = 4;

GREEDY RESULTS-PROFIT: 19713

TEST-4

P = 5; D=5; X= 30; T = 3; B=150; C = 4;

GREEDY RESULTS-PROFIT: 19175

TIME COMPLEXITY COMPARISON

FUNCTIONS	TIME COMPLEXITY
PART 1 DYNAMIC	O (N*(N+N)) = O(N^2)
PART 1 GREEDY	O (N)
PART 2 DYNAMIC	O (N*N*N) = O(N^3)
PART 2 GREEDY	O (X*(N+N)) = O(X*N)

X and N different array sizes. N is the investment list size. X is only a dimension which taken from 2D array to compare each company for each line.

SPACE COMPLEXITY COMPARISON

FUNCTIONS	SPACE COMPLEXITY
PART 1 DYNAMIC	O(N*N+1) = O(N^2)
PART 1 GREEDY	O (1)
PART 2 DYNAMIC	O(N*N+1) = O(N^2)
PART 2 GREEDY	O (N + 6(TEMP VAR)) = O(N)

There are six different variable and one array we used on part 2 greedy.

There is one variable and one 2D array which we hold the variables inside on part 2 dynamic.

We only used an integer variable to find minimum cost on part 1 greedy.

We used 2D array and just one variable to decide minimum cost in part 1 dynamic.

SCREENSHOTS