CSE 321 Homework 5 Report

Sema Köse

141044002

Course Assistant : M. Burak Koca

EXPLANATIONS OF THE QUESTIONS

Question 1:

I. Algorithm:

Let J be the array of jobs J = [[t_1, w_1]…[t_n, w_n]].

If we calculate the w_i / t_i for each job j_i and sort in decreasing order, jobs will be sorted in order to minimizing the weighted sum.

For example;

J = [[4, 7], [9, 4], [10, 9], [3, 8]]
J_1 = [4, 7] T_1 = 4 W_1 = 7
J_2 = [9, 4] T_2 = 9 W_2 = 4
J_3 = [10, 9] T_3 = 10 W_3 = 9
J_4 = [3, 8] T_4 = 3 W_4 = 8

W_1 / T_1 = 7 / 4 = 1.75
W_2 / T_2 = 4 / 9 = 0.4
W_3 / T_3 = 9 / 10 = 0.9
W_4 / T_4 = 8 / 3 = 2.7

Job Order = J_4 , J_1 , J_3 , J_2

Lets calculate;

$C_1 = T_4 * W_4 = 3*8 = 24$
$C_2 = (T_4 + T_1)* W_1 = 7*7 = 49$
$C_1 + C_2 = 73$

If we had chosen opposite:

$C_1 = T_1 * W_1 = 4*7 = 28$
$C_2 = (T_4 + T_1)* W_4 = 7*8 = 56$
$C_1 + C_2 = 84$
The total sum would be higher.

In conclusion, if we choose the job which has minimum t_i value according to maximum w_i value; we will be choose the minimum t_{i-1} for the next job j_i. In other words we will be increse the multiplier for the next weight with the possible smallest value.

II. Time Complexity:

We're using a bubble sort algorithm here for sorting the values due to a condition; which take $O(n^2)$ time.

$$T(n) \in O(n^2)$$

Question 2-a:

I. Proof:

As I calculated and printed with the code in the .py file :

```
for i= 1 to n
        if Ni < Si then
                Output “NY in Month i” else
                Output “SF in Month i” end
```

The algorithm above does not correctly solve the problem because if th e table was:

#	Month1		Month2		Month3	Month4
NY	1	3	2	6		
SF	50	20	20	4		

M = 10

It outputs:

['NY', 'NY', 'NY', 'SF']

The M = 10, so according to the algorithm the sum of the costs will be :

Total Cost = 1 + 3 + 2 + 4 + 10 = 20

But if we would be taken the NY as last city instead of SF,
we won't have to be add M value at the end
and the correct output would be:

['NY', 'NY', 'NY', 'NY']

And the sum of the costs would be:

Total Cost = 1 + 3 + 2 + 6 = 12

Question 2-b:

I. Algorithm:

Let $S[s_1 \ldots s_n]$ and $N[n_1 \ldots n_n]$ be two lists that represents operating costs of 2 cities N for NY and S for SF. n is the # of months and M is the moving cost.

M = 10;

ShiftCostBefore = 0;

#	Month1	Month2	Month3	Month4
NY	**1**	3	20	30
SF	**50**	20	2	4
Choosen	**1**	3	2	4
Month	**NY**	NY	SF	SF

First we take the smallest one of the **first costs;**

N[0] = 1 < S[0] = 50

Choosen[0] = 1 , Month[0] = NY

Then we compare the second costs with calculating the summation with the moving cost;

N[1] = 3 < ((ShiftCostBefore + M= 10) + (S[1] = 20) = 30)

Choosen[1] = 3 , Month[1] = NY

We keep comparing;

N[2] = 20 > (((ShiftCostBefore = 0)+(M= 10) = 10) + (S[2] = 2) = 12)

Choosen[2] = 2 , Month[2] = SF , ShiftCostBefore += M

And last comparison;

N[3] = 30 > (((ShiftCostBefore = 10)+(M= 10) = 20) + (S[3] = 2) = 22)

Choosen[3] = 2 , Month[2] = SF

We calculate shifts total so far, cause the next value cauld be greater than our moving cost.

II. Time Complexity:

For each record, we compare once so the algorithm tames O(n) time.

$$T(n) \in O(n)$$