CSE 321 Introduction to Algorithm Design

Fall 2019 – HW5

Name:Berke Süslü

Number:161044076

1)The recurrence relation is:

DP[i][j]= {citytable[0][0] if i=0 and j=0

citytable[1][0] if i=1 and j=0

min(DP[0][j-1]+citytable[0][j],DP[1][j-1]+citytable[1][j]+cost) if i=0

min(DP[1][j-1]+citytable[1][j],DP[0][j-1]+citytable[0][j]+cost) if i=1

i -> city

j->current day

cost->travel cost between cities.

I assume my citytable[0] is the NY city and citytable[1] is SF city.

The algorithm chooses the minimum cost; if the changing city is worth, changes the city and continues from there.

Time complexity: The algorithm runs in Θ(n) time because travels the arrays only once and the operations takes constant time.

2) The algorithm is from the our lecturer's note.

The optimal algorithm is joinning the session which finishs earliest.

Time complexity: The algorithm takes Θ(nlogn) time due to the sorting operation.(Mergesort)

4) The recurrence relation is:

DP[i][j]={-i*gap if i=0

-i*gap if j=0

Max(DP[i-1][j-1]+score(s1[i],s2[i]),DP[i-1][j]-gap,DP[i][j-1]-gap) if i≠0 and j≠0

Time complexity:O(m*n) (Psuedo-polynomial time) m-> size of first string

n-> size of second string

The algorithm is similiar to knapsack problem with dynamic programming.

5) The optimal algorithm is summing the smallest number with the second smallest number until all numbers are summed.

Time Complexity: T(n)= $\sum_{i=0}^{n-1} \sum_{j=0}^{n} 1$

T(n)=(n-1)*n=n^2-n

T(n)=Θ(n^2)