CO 322 Data Structures and Algorithms

Lab 02 - Dynamic Programming

E number:E/16/267

b).What is the runtime complexity of your implementation.

This is complexity result get from the java code;

From station 0 to station 1:
Minimum cost=3
Complexity=1

From station 0 to station 2:
Minimum cost=5
Complexity=3

From station 0 to station 3 :
Minimum cost=7
Complexity=9

From station 0 to station 4 :
Minimum cost=8
Complexity=27

From station 0 to station 5 :
Minimum cost=6
Complexity=81

From station 0 to station 6 :
Minimum cost=7
Complexity=243

In above we checked cost of paths from 0-(7-1).So the runtime complexity of the minCost function is exponential because it gives every possible path from 0 to n-1.Complexity varies by $T(n)=3^{n-1}$ (1,3,9,27,81,243). Here T(n) is the runtime complexity of nth problem and n=destination station. Therefore, we can say that runtime complexity is $O(3^n)$ because 1 is a constant in (n-1) .

(c) Argue that dynamic programming can be used to improve the runtime.

Dynamic programming is useful when recursive algorithm finds itself reaching the same situations (input parameters) many times. There is a general transformation from recursive algorithms to dynamic programming known as *memoization*, in which there is a table storing all results ever calculated by your recursive procedure. When the recursive procedure is called on a set of inputs which were already used, the results are just fetched from the table. Dynamic programming can be even smarter, applying more specific optimizations. For example, sometimes there is no need to store the entire table in memory at any given time.Therefore Dynamic programming can be used to improve runtime

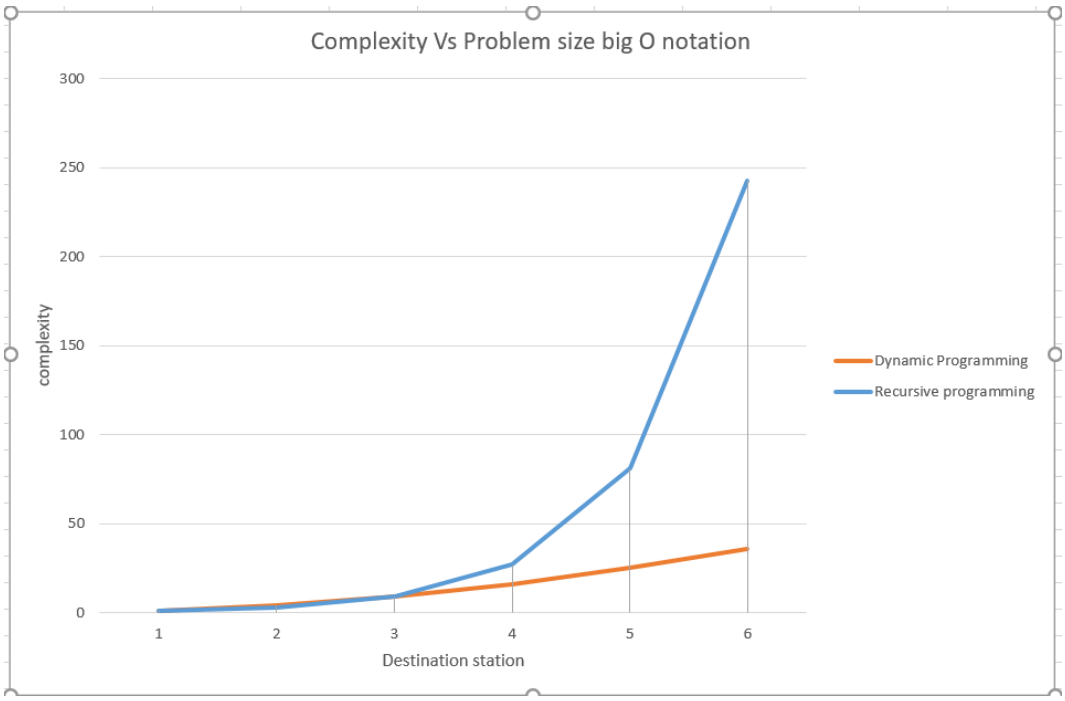

Figure :Complexity of Dynamic programming and Recursive programming for given mincost function

From the above figure also we can say that Dynamic programming is more effective than recursive programming for same problem size

e)Calculate the runtime of your implementation in part 4 above. Assume, hashing is O(1).

From station 0 to station 1
Minimum cost=3
Complexity=1

From station 0 to station 2
Minimum cost=5
Complexity=3

From station 0 to station 3
Minimum cost=7
Complexity=7

From station 0 to station 4
Minimum cost=8
Complexity=13

From station 0 to station 5
Minimum cost=6
Complexity=21

From station 0 to station 6
Minimum cost=7
Complexity=31

From the the above observation in minCostDynamicPro function,
$T(n)= n^2 - n + 1$ because we assumed that hashing takes constant time. Here T(n) is the runtime complexity of nth problem and n=destination station. This is a quadratic algorithm, therefore runtime complexity according to Big O notation is equal to $O(n^2)$.
So in dynamic implementation it gives better complexity than recursive implementation.