HW5

January 20, 2022

question 1:

Let's connect vertex s to all the other vertices with a weight of 0. now we run Bellman Ford to find the shortest path for s to all other vertices. the shortest path from s to v will have the weight of the shortest path from the best source to v. this is true because any path from $s \rightarrow u$ to v is will be the same as the path from u to v. as the weight from s to u is 0. Also, the path of $s \rightarrow v$ is equal to the path from v to itself.

Time complexity:

Connecting s to all the vertices - $O(V)$

Running Bellman Ford - $O(\mid V \mid * \mid E \mid)$

Total running time: $O(\mid V \mid * \mid E \mid)$

question 2:

Starting from vertex b we use Bellman Ford to find the shortest path to a. then we add the weight of the edge between a and b to create a cycle. the weight of the shortest path will be $d(b, a) + w(a, b)$.

Total running time: $O(\mid V \mid * \mid E \mid)$.

question 3:

we construct a new graph $G' = (V', E')$ in which for $V' = \{v' | v \in V\}$ and $E' = \{(u', v') | (u, v) \in E\}$ and $\forall (u', v') \in E'$ $w(u', v') = w(u, v) + w(v)$. we run Dijkstra on G' to find the shortest paths from any pair s' and t' in V', then for each path we subtract the weight of t' to get the shortest path without the weight for the source and target.

time complexity:

1. constructing the graph: $O(|V| + |E|)$.
2. Running Dijkstra: $O((|V| + |E|) log(|V|))$.

total running time: $O((|V| + |E|) log(|V|))$.

question 4:

let $\mid S_1 \mid = n$ and $\mid S_2 \mid = m$

first we create a table for all heaviest common subsequence HCS with a dimension of $(n+1),(m+1)$.

$HCS[i,j]$ will be the heaviest common subsequence for i letters in $S_1$and j letters in S_2.

We fill the table with 0's in row $i=0$ and column $j=0$.

Then we go over each row and fill the columns according to the following logic:

- $HCS[i,j] = HCS[i-1,j-1] + W(S_1[i])$ if $S_1[i] = S_2[j]$ then
- $HCS[i,j] = max(HCS[i-1,j], HCS[i,j-1])$ if $S_1[i] \neq S_2[j]$

where $HCS[0,j] = HCS[i,0] = 0$.

In case $S_1[i] = S_2[j]$ the last letter is included in the HCS which will be the HCS of the $i-1$ letters in $S_1$and the $j-1$ letters in S_2+ the weight of the last letter.

In case $S_1[i] \neq S_2[j]$. $S_1[i]$ so one of them or both is not included in the $HCS[i,j]$. so we need to check both cases and take the maximum of them, thus $max(HCS[i-1,j], HCS[i,j-1])$.

Going over the table will be in time complexity of $O(n*m)$.

question 5:

Let's define a matrix M of dimension $(n+1),(T+1)$. $M[i,j]$ will be the solution for the maximal number of items, out of i items, the robber can carry in a bag with maximum capacity of j.

Base case:

$M[0,j] = M[i,0] = 0$ for every i, j

General case:

$$M[i,j] = \begin{cases} M[i-1,j] & w_i > j \\ max(M[i-1,j], 1 + M[i-1, j-w_i\} & w_i \leq j \end{cases}$$

In order to find $M[i,j]$ we have two options:

1. we don't add w_i to the bag. in this case w_i is heavier that the remaining capacity, so our solution lies on the maximal number of items with same capacity but with $i-1$ items $\rightarrow M[i-1,j]$.
2. we add w_i to the bag, in this case our solution is the maximum between option 1 and option 2 which now its solution lies on solving a problem to find the maximal number of items with capacity minus w_i and with $i-1$ items, + 1, because we added the item to the bag.$\rightarrow max(M[i-1,j], 1 + M[i-1, j-w_i\}$.

we begin filling the table first with zeros for the base cases and then for each row, we fill it using the previous row values according to the algorithm.

Time complexity: $O(n*T)$.