Ryan Pollack

rep127

I have not discussed this exam with anyone except the professor and the TAs of CS 344. I have not used any online resources during the exam except for those accessible from the Canvas website.

Problem 1

Part 1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S | A | E | D | C | F | B |
| 0 | 3 | 4 | 7 | 8 | 9 | 14 |

Part 2:

F 🡪A🡪D🡪B🡪C🡪E🡪G

F 🡪A🡪D🡪B🡪C🡪G🡪E

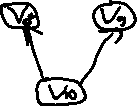
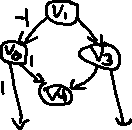
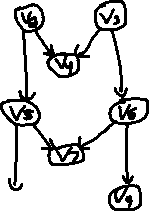
Problem 2

Part 1:

If P is the shortest x-y path in G then it is also a shortest x-y path in G’ and vice versa

For every pair of vertices(x,y) E G, DIST’(x,y) = DIST(x,y) + (x) + (y)

Part 2:



The maximum possible w’(x,y) is 2 for the edge (v10, v8)

Problem 3:

1. T[i][j] is true of there exists S’ C\_ {si, …sn}such that and count of red elements 100
2. DP relation T[i][j] is true if T[i-1, j] is true and count of red elements is 100
3. T[i][j] = 0
4. Return true if T[i][j] 100 red Elements else return false
5. Pseudocode:

subsetSumWithRedElements(int arr[], int n, int sum)

Create boolean 2D array with 2 rows and sum + 1 columns

For i = 0 until i <= n while i++ every loop

For j = 0 until j <= sum while j++ every loop

If i = 0

Set T[i % 2][j] = false

Else If j = 0

Set T[i % 2][j] = true

Else if arr[i - 1] <= j

//set current index based on the true or false values

Set T[i % 2][j] to T[(i + 1) % 2][j - arr[i - 1]] OR T[(i + 1) % 2][j]

Else

T[i % 2][j] = T[(i + 1) % 2][j]

End Else If

End For

End For

Let counter = 0

For loop through T starting at T[n % 2][sum] working backwards

Go up one row

If value is true

Check if color is Red if so add 1 to counter

Go up one row

Else

Go up one row and go back sum amount of columns

End Else If

End For

If counter <= 100

return true

else

return false

End Function

Problem 4

1. T[n] is the number of paths from the source vertex to the destination vertex in DAG G where the number of edges is 100
2. DP relation: Total paths is 100 edges for all paths from vertex s to destination t
3. T[n] = 0
4. Sum of T[n]
5. Return Pseudocode:

The following is a modified version of the BFS algorithm which runs in 0(|E|) time complexity

Create a queue which will store path(s) to process

Initialize the queue with first path starting from src

Create new array T[n] to hold all of the valid paths from src to dest

Loop through queue till is not empty

Get the frontmost path from the queue

check if the last node of this path is the same as the dest

If the length of the path - 1 <= 100 then

Add path to T[n]

Else

Break out of the loop

End If

Loop through all the vertices connected to the current vertex

If the vertex is not visited in current path

Create a new path from earlier path and append this vertex

Insert this new path to queue

End IF

Return the size of T[n]

Problem 5:

I don’t know