A. C. Patil College of Engineering Kharghar Navi-Mumbai Maharashtra

Name : Priyush Bhimrao Khobragade

Roll NO: 52
PRN NO :211112018

Subject : Analysis of Algorithms (AOA)
Assignment-03

[ADA]

Assignment No-03

Q.1) Find the single source shortest path for a given graph assume node i as a source vertex. (Use DIJKSTRA's Algorithm).

→ Dijkstra algorithm proposed by E.W. Dijkstra to solve a single source path problem with a positive weighted ~~graph~~ connected graph.

It describe a multistage progressive solution that computes the shortest paths from a source node to only other nodes in the given graph.

• Algorithm

Input : w[1:n, 1:n)

weight cost/cancel matrix of a connected graph G = (V, E) V → Vertex, E → edge, n = |V| = No. of vertex in a graph.

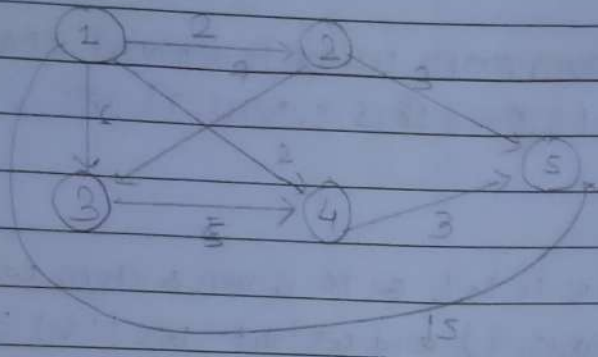
Output : dist [1:n] given shortest distance of each node i from the source node v_0 & pred [1:n] give the predecessor of each node i. in path from v_0 to i, $i < i \leq n$ */

```
{
  F = ϕ   :        // initialization
  for (i!=1 ; i<=n ; i++)
  {
  pred [i] = ENULL ;
  dist [i] = INFINITY ;
  }
```

```
dist[v0] = 0; prev[v0] = v0;   // Set value for source
for(i=1; i<n; i++)
{
    insert node i into min-priority queue according
    to its dist[i] value;
}
while (a priority queue is non-empty)
{
u := delete node from a priority queue ..

F: F ∪ {u}                    // Add u in final solution set

for (each node <u, v_i> ∈ E and v_i ∈ V-F)
{
    if dist[v_i] > dist[u] + wt[u, v_i]
                              // Relaxation on edge (u, v_i)
    {
    dist[v_i] = dist[u] + wt[u, v_i]
                              update dist[v_i]
    pred[v_i] = u;
    }
}
}
}
```



Sol?

This graph is directed graph positive weight. So we apply Dijkstra's algorithm to enumerate all shortest path from a source node (say a 1 to all other vertex 2, 3, 4, 5 as below

- We use formula:

dist[vi] := dist[u] + cost[u, vi]

pred[vi] := u; ... if dist[vi] > dist + wt(u, vi)

Interation	F (Final sol^n Vertex	Selected	dist[vi], pred[vi]				
in		vertex u	vi: 1	2	3	4	5
initialization	ϕ	1	(0, 1)	(∞, Nil)	(∞, Nil)	(∞, Nil)	(∞, Nil)
1.	{1}	2	(0, 1)	(2, 1)	(6, 1)	(12, 1)	(5, 1)
2.	{1, 2}	5	(0, 1)	(2, 1)	(6, 1)	(12, 1)	~~(5, 1)~~ (5, 2)
3	{1, 2, 5}	3	(0, 1)	(2, 1)	(6, 1)	(12, 1)	(5, 2)
4	{1, 2, 5, 3}	4	(0, 1)	(2, 1)	(6, 1)	(11, 3)	(5, 2)
5.	{1, 2, 5, 4, 3}	--	(0, 1)	(2, 1)	(6, 1)	(11, 3)	(5, 2)

Path	Shortest distance
1 → 2 : 1 - 2	2
1 - 3 : 1 - 3	6
1 - 4 : 1 - 3 - 4	11
1 - 5 : 1 - 2 - 5	5

2) Find maximum profit using fractional knapsack approach
$n=6$, $m=13$, $P=\{18,5,9,10,12,7\}$, $w=\{7,2,3,5,3,2\}$

Sol: Let $i_1, i_2, i_3, i_4, i_5, i_6$ be the given 6 item with profits $p(1:6)$ $=\{18,5,9,10,12,7\}$ and weight $w(1:6)=\{7,2,3,5,3,2\}$

- To solve this knampsack instance by the greedy method we first calculate the ratio p_i/w_i; $1 \le i \le n$.

Item(i)	p_i	w_i	p_i/w_i
1	18	7	2.57
2	5	2	2.5
3	9	3	3
4	10	5	2
5	12	3	4
6	7	2	3.5

- We arrange the given item in descending order of p_i/w_i as i_5, i_6, i_3, i_1, i_2 and i_4.

- Consider the available capacity $m=13$ of a knapsack we can add item i_5, i_6, i_3, i_1, i_2 as a whole

∴ The remaining capacity of a knapsack
$= 13-(3+2+3+7+2) = -4$

Page No
Date

Due to insufficient capacity of a knapsack item 4 is added fractional part = 1/5 67 giving profit

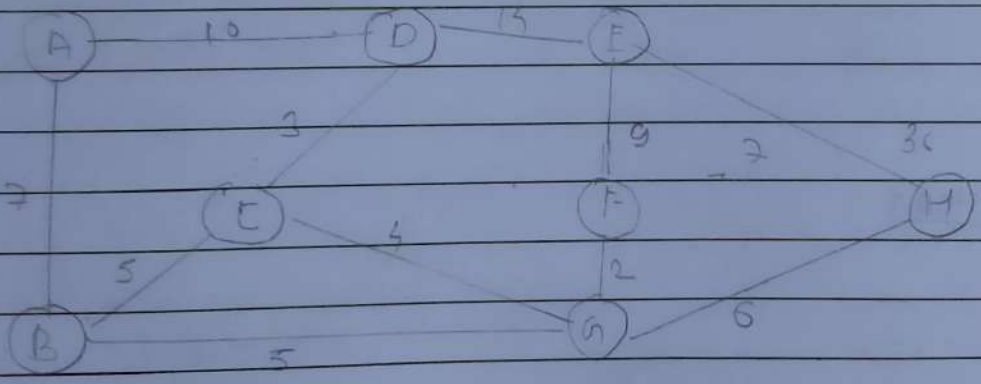
=

- Method i select only with max profit

Obj	P_i	w_i	remaining value
1	18	7	13 − 7 = 6
2	12	3	6 − 3 = 3
3	3×2/3	3	3 − 3 = 0

Hence, the max profit is = 36.

Q.3) Find the minimum spanning tree for the given graph using prims and Kruskals Algorithm



Page No.
Date

Prim's Algorithm :-

Step 1 : Select any connected vertex with minimum weight

step 2 : unvisite vertex which is adjecent of visited vertices with minimum weight

Step3 : Repeat step2 untill all vertices are visited

Step 1)

S (Select) = { a }

VIS (Remain) = { B, C, D, E, F, G, H }

A = { }

light edge { A, B }

Step 2)

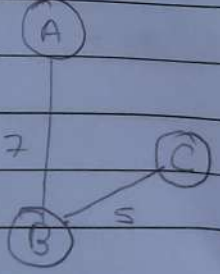
S = { a, b }

VIS = { C, D, E, F, G, H }

A = { (A, B) }

light Edge { (B, C), (A, D) }

Step 3



S = { a, b, c }

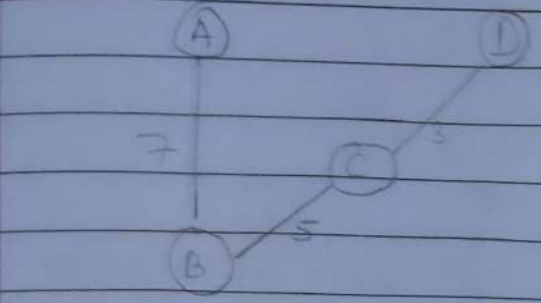
VIS { D, E, F, G, H }

A = { { A, B }, { B, C } }

light Edge { (C, D), (C, G) }

Page No.
Date

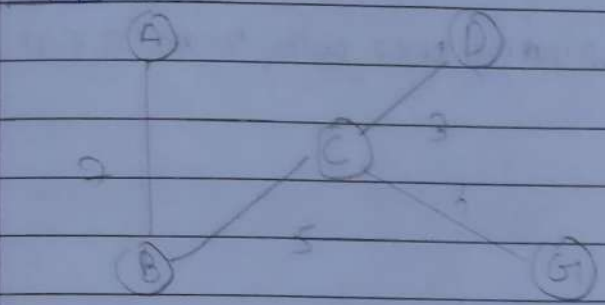
Step 4 :-



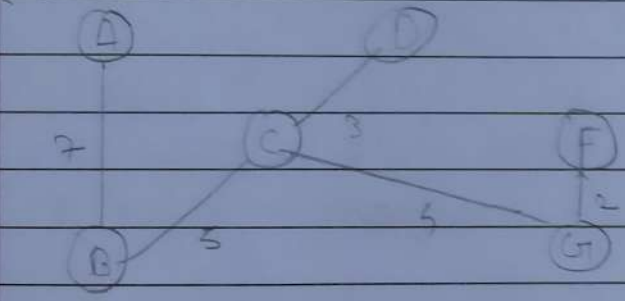
S = { A, B, C, D }

V/S = { E, F, G }

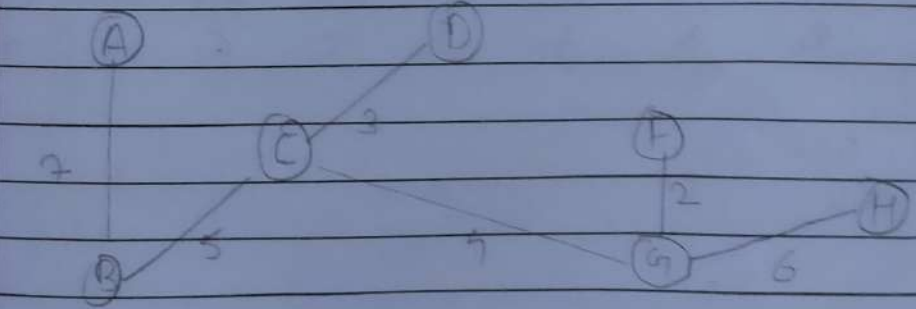
Step 5 :-



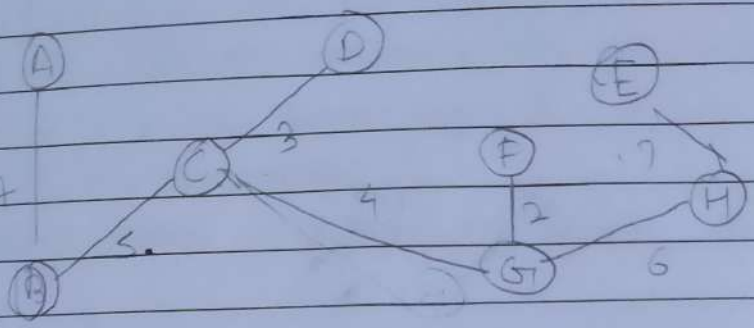
Step 6



Step 7



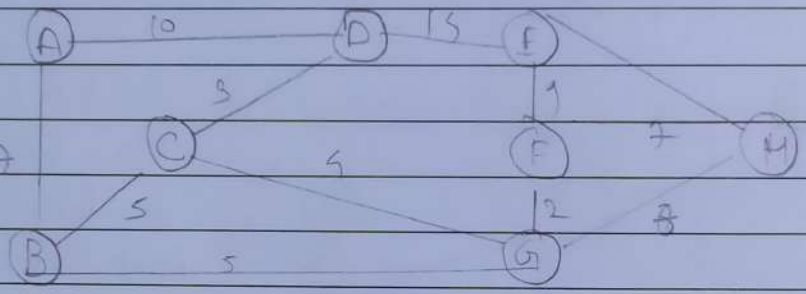
Step: 8



Hence the minimum spanning tree with vertex & cost is
34

- Kruskal Algorithm:-

Step(1) Remove all loop & parallel edge.



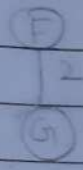
Step(2): Arrange all edge in their increasing order

edge	GF	CD	CH	BC	BH	GH	AB	EF	AD
	2	3	4	5	5	6	7	9	10

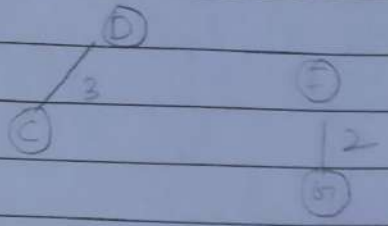
DE HE
15 7

Page No
Date

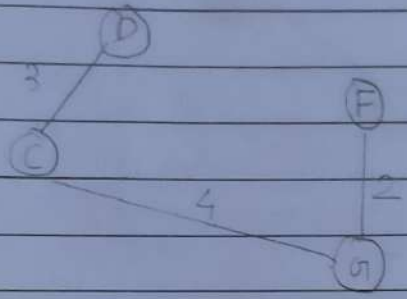
Step(2) :- Add the edge which has least weightage



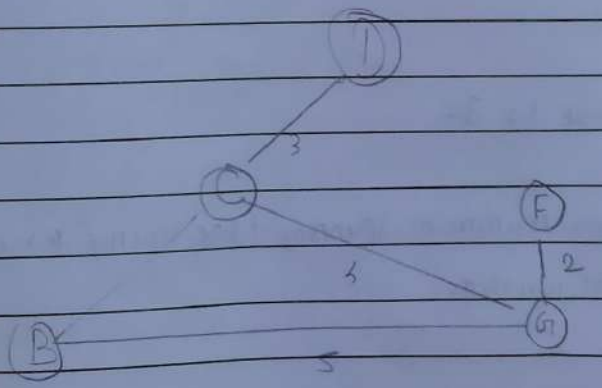
Next cost is 3 & the edge is CD.



Next cost is 4 & the edge is CG

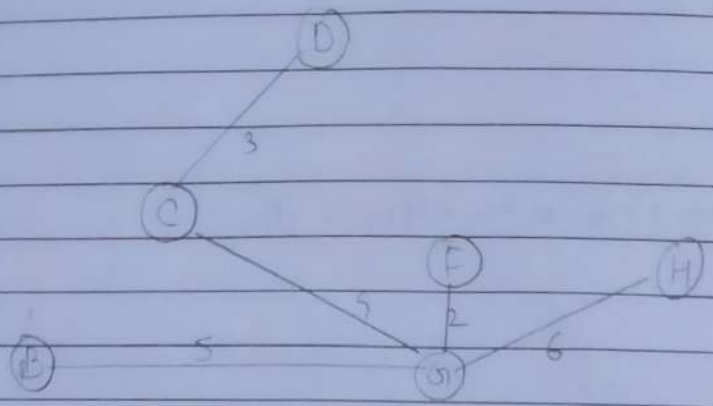


Next cost is 5 and the edge is BC, BG But it we take cost 5 BC it create cyclic then it ignore or discard

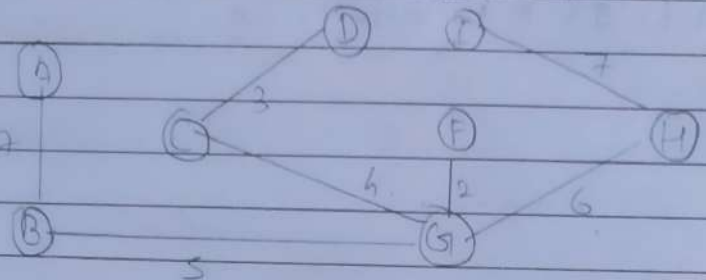


Page No
Date

· Next cost is 6 & the edge is GH.



· Next cost is 7 & edge is AB & EH



· Next cost 9, 10, & 14 & edge AD, DE if we has select cyclic from Hence is discard.

The minimum cost is 34.

Hence this Graph minimum spanning tree using Kruskal's Algorithm.