UNIT-3

Greedy Method

- Knapsack problem (fractional)
- Minimum cost spanning tree
  - Kruskal
  - Prims
- single source shortest path
  - Dijikstra
  - Bellman ford

Strassen's Matrix Multiplication :-

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$= \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$$

8 multiplications
4 additions

Strassen's → 7 multiplications
→ 18 addition | substractions

Normal matrix multiplication

$$T(n) = \begin{cases} \theta(1), & \text{if } n \le 1 \\ 8T(n/2) + n^2, & \text{if } n > 1 \end{cases}$$

8 → no. of mul. operators

n^2 → complexity for addition

$O(n^3)$

Strassen's matrix multiplication

$$T(n) = \begin{cases} \theta(1), & \text{if } n \le 1 \\ 7T(n/2) + n^2, & \text{if } n > 1 \end{cases}$$

7 → no. of mul. operators

n^2 → complexity for addition

$O(n^{2.81})$

$P = (A_{11} + A_{22})(B_{11} + B_{22})$

$Q = (A_{21} + A_{22})B_{11}$

$R = A_{11}(B_{12} - B_{22})$

$S = A_{22}(B_{21} - B_{11})$

$T = (A_{21} + A_{11})B_{22}$

$U = (A_{21} - A_{11})(B_{11} + B_{12})$

$V = (A_{12} - A_{22})(B_{22} + B_{21})$

$C_{11} = P + S - T + V$

$C_{12} = R + T$

$C_{21} = Q + S$

$C_{22} = P + R - Q + U$

Q:- $A = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$, $B = \begin{bmatrix} 8 & 9 \\ 6 & 7 \end{bmatrix}$

$P = (A_{11} + A_{22})(B_{11} + B_{22})$

$= (2+5)(8+7)$

$= 7 \times 15 = 105$

$Q = (A_{21} + A_{22}) B_{11}$

$= (4+5) 8$

$= 72$

$R = A_{11}(B_{12} - B_{22})$

$= 2(9-7)$

$= 4$

$S = A_{22}(B_{21} - B_{11})$

$= 5(6-8)$

$= -10$

$T = (A_{12} + A_{11}) B_{22}$

$= (3+2) 7$

$= 35$

$U = (A_{21} - A_{11})(B_{11} + B_{12})$

$= (4-2)(8+9)$

$= 34$

$V = (A_{12} - A_{22})(B_{22} + B_{21})$

$= (3-5)(7+6)$

$= -26$

$C_{11} = P + S - T + V$

$= 105 \cancel{+} -10 - 35 - 26$

$= 34$

$C_{12} = R + T = 4 + 35 = 39$

$C_{21} = Q + S = 72 - 10 = 62$

$C_{22} = P + R - Q + U = 105 + 4 - 72 + 34$

$= 71$

GREEDY STRATEGY :-

- It is an algorithm designing technique.
- A greedy method makes a choice that looks better at a moment.

Greedy

feasible solution   optimal solution

- A program problem having 'n' I/P & require us to obtain a subset that satisfies some constraints.
  Any subset that satisfies the constraints is called feasible soln.
- Now, we need to find a feasible soln. that either maximize or minimize the given objective function.
  A feasible soln. that does this is called optimal soln.
- In greedy method, at each stage decision is made whether particular I/P is in optimal solution.

Knapsack Problem (fractional)

Knapsack ↓ Bag

- We are given 'n' objects & a knapsack (bag).
- Object i has weight 'w_i' & knapsack has capacity 'm'.
- If a fraction x_i where $0 \le x_i \le 1$ of object 'i' is placed in knapsack, then profit $P_i x_i$ is earned.
- So, main objective is to fill a knapsack that maximize the total profit earned.

formally problem can be stated as :-

maximizes $\sum_{1 \le i \le n} P_i x_i$ ——— ①

subject to $\sum_{1 \le i \le n} w_i x_i \le m$ ——— ②

and $0 \le x_i \le 1$, $1 \le i \le n$ —— ③

Imp

- A feasible soln. is any set $(x_1, x_2, --- x_n)$ that satisfies eq ② & ③.
- An optimal soln. is a feasible soln. that satisfies eq ①

Greedy Knapsack (m, n)

// calculate $\frac{P_i}{W_i}$ such that $\frac{P_i}{W_i} > \frac{P_{i+1}}{W_{i+1}}$

```
{
  for i=1 to n do x[i] = 0.0   float value
                  solution vector
    U = m
  for i = 1 to n do
  {
    if (W[i] > n) then break;
      x[i] = 1.0 , U = U - W[i]
  }
  if (i ≤ n) then x[i] = U / W[i];
}
```

Q:- $n = 3$, $m = \cancel{15}\ 20$

$P_1, P_2, P_3 = 25, 24, 15$

$W_1, W_2, W_3 = 18, 15, 10$

$\frac{P_1}{W_1} = \frac{25}{18} = 1.388$ $\quad \frac{P_2}{W_2} = \frac{24}{15} = 1.6$ $\quad \frac{P_3}{W_3} = \frac{15}{10} = 1.5$

Arrange in descending order

$\frac{P_2}{W_2} = 1.6$ $\quad \frac{P_3}{W_3} = 1.5$ $\quad \frac{P_1}{W_1} = 1.3$

for i=1 to 3 do

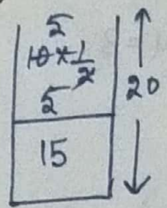
if (W[1] > 15)

$x[1] = 1$, $u = 20 - w[1]$
$= 5$

if $(w[2] > 5)$
$(10 > 5)$ then break,

if $(2 \le 3)$ then $x[2] = 5/10 = 1/2$
$\{1, 1/2, 0\}$

$\boxed{\text{Optimal soln.} - \{0, 1, 1/2\}}$ → arrange in normal order



Max. profit $= (0 * 25 + 24 * 1 + 15 * 1/2)$
$\boxed{\text{Max. profit} = 31.5}$

Q:- $n = 7$, $m = 15$
$\{P_1, P_2, P_3, --- P_7\} = \{10, 5, 15, 7, 6, 18, 3\}$
$\{w_1, w_2, w_3, --- w_7\} = \{2, 3, 5, 7, 1, 4, 1\}$
find soln. vector & max. profit.

$\frac{P_1}{w_1} = \frac{10}{2} = 5$ $\quad \frac{P_2}{w_2} = \frac{5}{3} = 1.6$ $\quad \frac{P_3}{w_3} = \frac{15}{5} = 3$ $\quad \frac{P_4}{w_4} = \frac{7}{7} = 1$

$\frac{P_5}{w_5} = \frac{6}{1} = 6$ $\quad \frac{P_6}{P_6} = \frac{18}{4} = 4.5$ $\quad \frac{P_7}{w_7} = \frac{3}{1} = 3$

Arrange in descending order:-

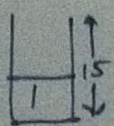
$\frac{P_5}{w_5} > \frac{P_1}{w_1} > \frac{P_6}{w_6} > \frac{P_3}{w_3} > \frac{P_7}{w_7} > \frac{P_2}{w_2} > \frac{P_4}{w_4}$

$6 > 5 > 4.5 > 3 > 3 > 1.6 > 1$

$x[i] = \{0, 0, --- 0\}$

for i=1 to 7
$m = 15$
$u = 15$

for i=1 to 7 do
if $(w[i] > 15)$
$(1 > 15)$ false
$x[1] = 1$, $u = 15 - 1 = 14$

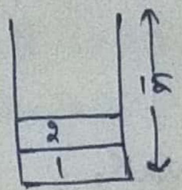


$i=2$

if $(w[2] > 14)$

$(2 > 14)$ false

$x[2] = 1$, $u = 14 - 2 = 12$

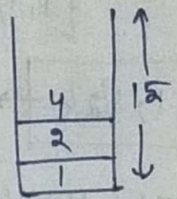


$i=3$

if $(w[3] > 12)$

$(4 > 12)$ false

$x[3] = 1$, $u = 12 - 4 = 8$

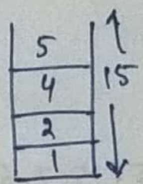


$i=4$

if $(w[4] > 8)$

$(5 > 8)$ false

$x[4] = 1$, $u = 8 - 5 = 3$

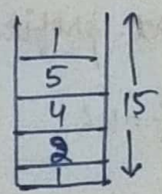


$i=5$

if $(w[5] > 3)$

$(1 > 3)$ false

$x[5] = 1$, $u = (3 - 1)$
$= 2$

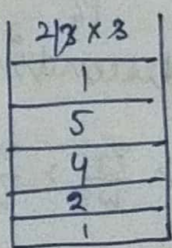


$i=6$

if $(w[6] > 2)$

$(3 > 2)$ true, then break

if $(6 \le 7)$, then $x[6] = 2/3 = 0.6$

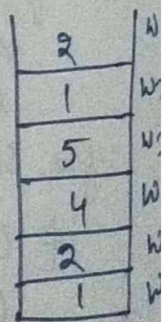


Soln:-

$\boxed{x[i] = \{1, 2/3, 1, 0, 1, 1, 1\}}$ → optimal soln.

Max. profit :- $= \{1 \times 10 + \frac{2}{3} \times 5 + 1 \times 15 + 0 \times 7 + 1 \times 6 + 1 \times 18 + 1 \times 3\}$

$\boxed{= 55.33}$



Q:- $n=4$, $m=21$

$w_1, w_2, w_3, w_4 = 5, 6, 9, 4$

$P_1, P_2, P_3, P_4 = 24, 25, 18, 15$

$$\frac{P_1}{w_1} = \frac{24}{5} = 4.8$$

$$\frac{P_2}{w_2} = \frac{25}{6} = 4.16$$

$$\frac{P_3}{w_3} = \frac{18}{9} = 2$$

$$\frac{P_4}{w_4} = \frac{15}{4} = 3.75$$

$$\frac{P_1}{w_1} > \frac{P_2}{w_2} > \frac{P_4}{w_4} > \frac{P_3}{w_3}$$

i ↓ 1, ↓ 2, ↓ 3, ↓ 4

$\boxed{u = 21}$

for $i = 1$ to 4

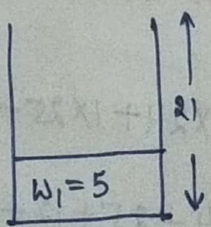
i) $i=1$

if $(w[1] > u)$

$(5 > 21)$ false

$x[1] = 1$, $u = 21 - 5$

$u = 16$



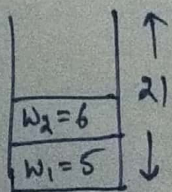
ii) $i=2$

if $(w[2] > 16)$

$(6 > 16) \rightarrow$ false

$x[1] = 1$, $u = 16 - 6$

$= 10$



Scanned by CamScanner

iii) $i=3$

if $(w[3]>u)$

$(4>10)$ false

$\boxed{x[3]=1}$

$u=10-4$

$\boxed{u=6}$

$w_4=4$
$w_2=6$
$w_1=5$

iv) $i=4$

if $(w[4]>u)$

$(9>6)$ true

then break

$(4\le 4)$ then,

$x[4] = 2\cancel{6}/\cancel{9}3 = 2/3$

$w_3=2/\cancel{3}\times\cancel{9}^{3}=6$
$w_4=4$
$w_2=6$
$w_1=5$

$\boxed{x[i]=\{1,1,2/3,1\}}$

optimal solution

Max. profit $=\{1\times 24+1\times 25+ 2/\cancel{3}\times\cancel{18}^{6}+1\times 15\}$

$=(24+25+12+15)$

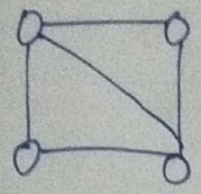
$\boxed{=76}$

Scanned by CamScanner

MINIMUM COST SPANNING TREE

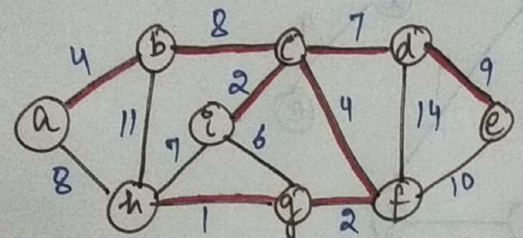
Let $G = \{V, E\}$ be an undirected connected graph. A subgraph $t = \{V, E'\}$ of G is a spanning tree iff t is a tree.

Ex



We have 2 methods to generate min. cost spanning tree

1) Kruskal (forest)
2) Prime (single tree)



Step 1: Make individual set of each vertex

A – {{a} {b} {c} {d} {e} {f} {g} {h} {i}}

Step 2: Arrange in ascending order

h – g → 1 ✓
g – f → 2 ✓
i – c → 2 ✓
a – b → 4 ✓
c – f → 4 ✓
i – g → 6 X
h – i → 7 X
c – d → 7 ✓
b – c → 8 ✓
a – h → 8 X
d – e → 9 ✓
e – f → 10 X
b – h → 11 X
d – f → 14 X

A= {{a}{b}{c}{d}{e}{f}{g}{h}{i}}

= {{h,g}, {a}, {b}, {c}, {d}, {e}, {f}, {i}}

= {{f,h,g}, {a}, {b}, {c}, {d}, {e}, {i}}

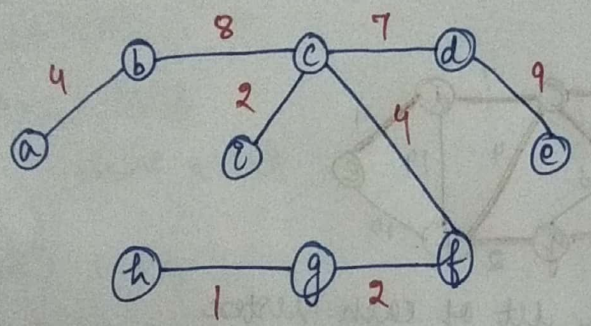
= {{f,h,g}, {a,b}, {i,c}, {d}, {e}, ~~{i}~~}

= {{i,c,f,h,g}, {a,b}, {d}, {e}, ~~{i}~~}

= {{i,c,d,f,h,g}, {a,b}, {e}}

= {{f,g,h,i,c,d,a,b}, {e}}

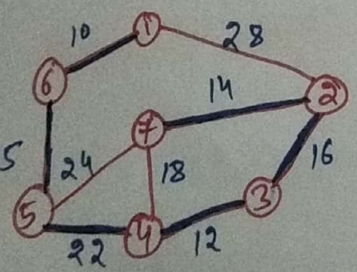
= {f,g,h,i,c,d,a,b,e}



Min cost = 4+8+7+9+2+4+1+2

$\boxed{=37}$

Q:-



1-6 → 10 ✓
3-4 → 12 ✓
2-7 → 14 ✓
2-3 → 16 ✓
4-7 → 18 ✗
4-5 → 22 ✓
5-7 → 24 ✗
5-6 → 25 ✓
1-2 → 28 ✗

A = {{1}, {2}, {3}, {4}, {5}, {6}, {7}}

= {{1,6}, {2}, {3}, {4}, {5}, {7}}

= {{1,6}, {2}, {3,4}, {5}, {7}}

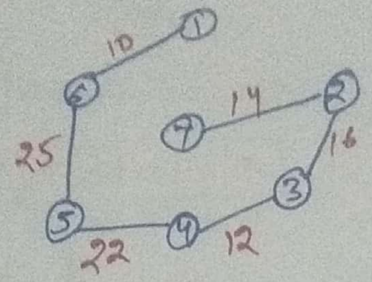
= {{1,6}, {2,7}, {3,4}, {5}}

= {{1,6}, {2,3,4,7}, {5}}

= {{1,6}, {2,3,4,5,7}}

1
10
12
14
16
22
25

$= \{\{1, 6, 2, 3, 4, 5, 7\}\}$



$$\text{Min. cost} = 10+12+14+16+22+25$$
$$\boxed{=99}$$

Kruskal's MST :-

Algo :-

1) $A \leftarrow \phi$
2) for each vertex $v \in V(G)$.
3) do make-set (v)
4) sort the edges of E in non-decreasing order by weight w.
5) for each edge $(u, v) \in E$ taken in non-decreasing order by weight
6) do if find set $(u) \neq$ find set (v)
7) $A \leftarrow A \cup (u, v)$
8) union (u, v)
9) return A.

PRIMS :-

- Prims algo. has a property that edges in set A always form a tree.
- Tree starts from root vertex R & grows until tree spans all vertices.
- At each step, light edge is added to tree.
- This strategy is greedy since tree is augmented at each step with an edge that contributes minimum amount to tree weight.

Algo :-

MST_PRIM(G, W, r)

1) for each u∈V[G]

2) do key [u] ← ∞

3) π[u] ← NIL

4) Key [r] ← 0

5) Q ← V[G]

6) while Q ≠ ∅

7) do u ← extract min (Q)

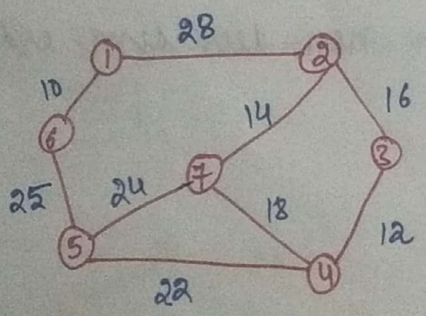
8) for each edge v ∈ adj(u)

9) do if v∈Q and w(u,v) < Key [v]

10) π [v] ← u

11) Key[v] ← w[u,v]

Q:-



Q =

0	∞	∞	∞	∞	∞	∞
1	2	3	4	5	6	7

extract (1)

Adj(1) ∈ 2, 6

do if 2 ∈ Q & w(1,2) < Key [2]

yes and 28 < ∞

true

Key [2] ← w [1,2] ⇒ 28

28	∞	∞	∞	∞	∞
2	3	4	5	6	7

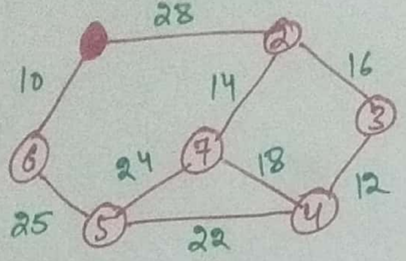
(Remove 1)

do if $6 \in Q$ and $w(1,6) < Key[6]$

yes & $10 < \infty$ true

$Key[6] \leftarrow w(1,6) = 10$

2	3	4	5	6	7
28	∞	∞	∞	10	∞



Extract(6)

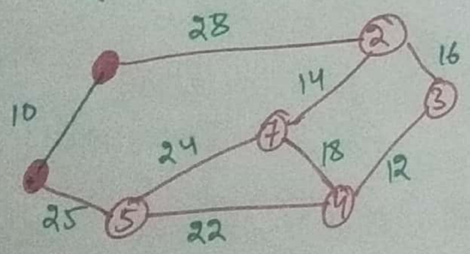
Adj(6) ∈ 5

do if $5 \in Q$ and $w(6,5) < Key[5]$

yes & $25 < \infty$ (true)

$Key[5] \leftarrow w[6,5] = 25$

2	3	4		5	7
28	∞	∞	~~∞~~	25	∞



Extract(5)

Adj(5) ∈ 7, 4

do if $7 \in Q$ and $w(5,7) < key[7]$

yes & $24 < key[7]$ (true)

$key[7] \leftarrow w[5,7] = 24$

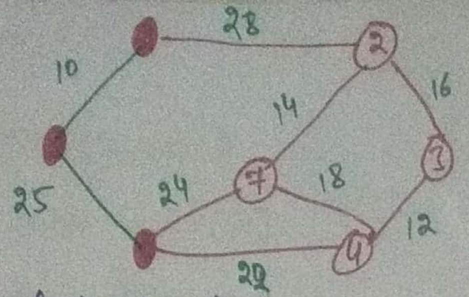
2	3	4	5	7
28	∞	∞	25	24

do if $4 \in Q$ & $w(5,4) < Key[4]$

yes & $22 < Key[4]$ (true)

$Key[4] \leftarrow w[5,4] = 22$

2	3	4	7
28	∞	22	24



Extract (4)

Adj(4) $\in$ 7, 3

do if $3 \in Q$ & $w(4,3) < Key[3]$

yes & $12 < \infty$ (true)

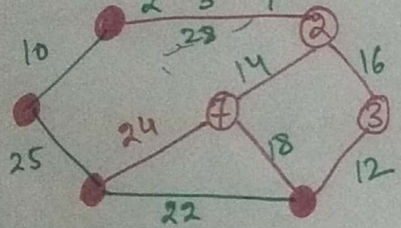
$Key[3] \leftarrow w(4,3) = 12$

do if $7 \in Q$ & $w(4,7) < Key[7]$

yes & $18 < 24$ (true)

$Key[7] \leftarrow w(4,7) = 18$

28	12	18
2	3	7



Extract (3)

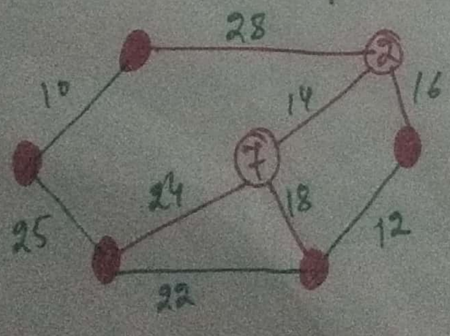
Adj(3) $\in$ 2

if $2 \in Q$ and $w(3,2) < Key[2]$

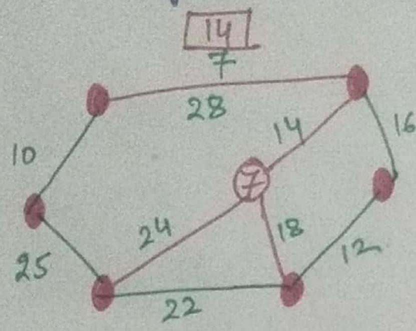
yes & $16 < 28$ (true)

$Key[2] \leftarrow w(3,2) \leftarrow 16$

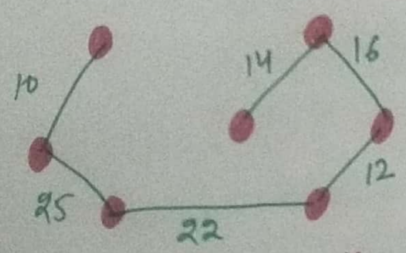
16	18
2	7



extract(2)
adj(2) ∈ 7
do if 7 ∈ Q and w(2,7) < Key[7]
yes & 14 < 18 (true)
Key[7] ← 14

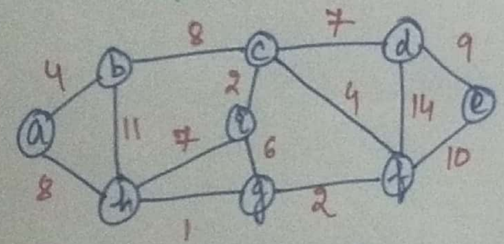


extract(7)



Min.cost = 10+14+16+12+22+25
$\boxed{=99}$

Q:-



Construct MST from above graph by considering 'a' as source vertex.

Q =	0	∞	∞	∞	∞	∞	∞	∞	∞
	a	b	c	d	e	f	g	h	i

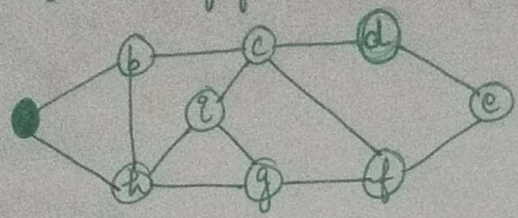
extract(a)
adj(a) ∈ b, h
do if b ∈ Q & w(a,h) < Key[h]
yes & 8 < ∞ (true)
Key[h] ← w(a,h) = 8

∞	∞	∞	∞	∞	∞	8	∞
b	c	d	e	f	g	h	i

do if h ∈ Q & w(a,b) < Key[b]
yes & 4 < ∞ (true)

Key[b] ← w(a,b) = ~~8~~ 4

b	c	d	e	f	g	h	i
~~8~~ 4	∞	∞	∞	∞	∞	8	∞



extract (b)

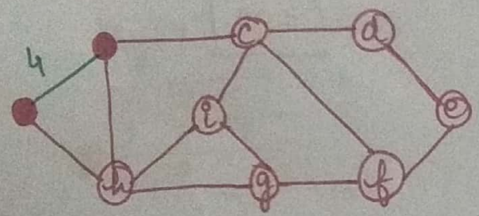
adj(b) ∈ c

do if c ∈ Q and w(b,c) < Key[c]

yes & 8 < ∞ (true)

Key[c] ← 8

c	d	e	f	g	h	i
8	∞	∞	∞	∞	8	∞



extract (c)

adj(c) ∈ i, d, f

do if i ∈ Q and w(c,i) < Key [i]

yes & 2 < ∞ (true)

Key[i] ← 2

d	e	f	g	h	i
∞	∞	∞	∞	8	2

adj (c) ∈ d

do if d ∈ Q and w(c,d) < Key[d]

yes & 7 < Key[d] (true)

Key[d] ← w(c,d) = 7

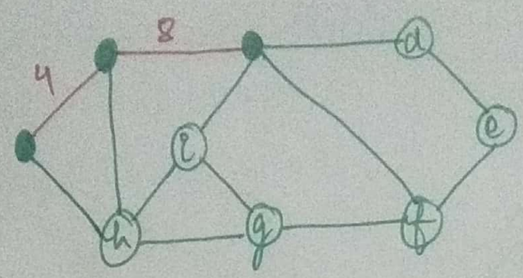
d	e	f	g	h	i
7	∞	∞	∞	8	2

do if f ∈ Q and w(c,f) < Key[f]

yes and 4 < Key[f] (true)

Key[f] ← w(c,f) = 4

7	∞	4	∞	8	2
d	e	f	g	h	i



Extract (i)

adj (i) ∈ h, g

do if h ∈ Q and w(i,h) < key[h]

yes and 7 < 8 (true)

key[h] ← w(i,h) = 7

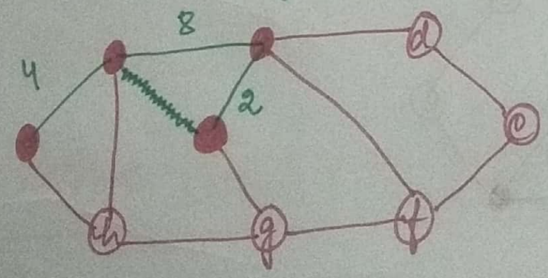
7	∞	4	∞	7
d	e	f	g	h

do if g ∈ Q and w(i,g) < key[g]

yes and 6 < ∞ (true)

key[g] ← w(i,g) = 6

7	∞	4	6	7
d	e	f	g	h



Extract (f)

adj(f) ∈ g, e

do if g ∈ Q and w(f,g) < key[g]

yes & 2 < 8 (true)

key[g] ← w(f,g) = 2

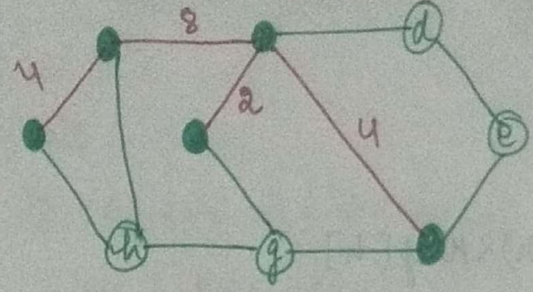
7	∞	2	7
d	e	g	h

do if e∈Q and W(f,e) < Key[e]

yes & 10 < ∞ (true)

Key[e] ← W(f,e) = 10

7	10	2	7
d	e	g	h



Extract (g)

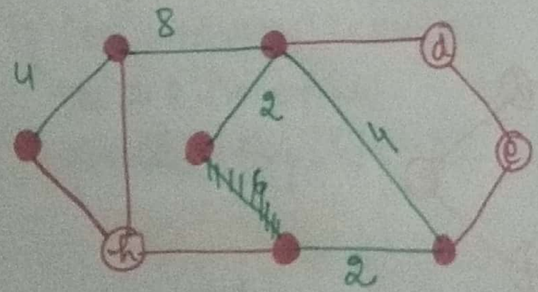
do adj(g) ∈ h

do if h∈Q & W(g,h) < Key[h]

yes & 1 < 7 (true)

Key[h] ← W(g,h) = 1

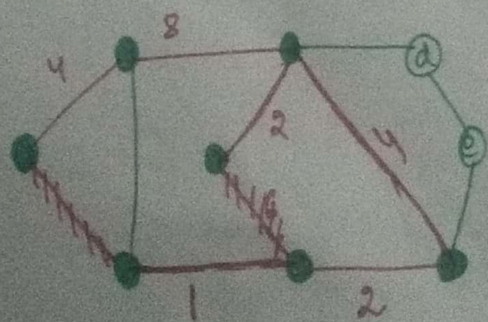
7	10	1
d	e	h



Extract (h)

adj(h) ∈ φ

7	10
d	e



Extract (d)

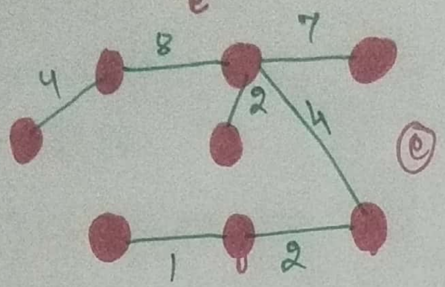
adj (d) ∈ e

do if e ∈ Q & w(d,e) < Key(e)

yes & 9<10 (true)

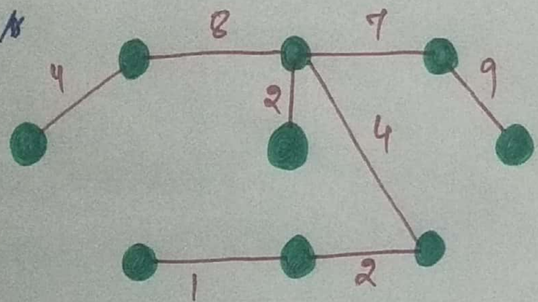
Key[e] = w(d,e) = 9

$\boxed{9}$
e



Extract (e)

adj (e) ∈ φ



Min. cost = 4+8+7+2+2+4+1+9

$\cancel{\boxed{=28}}$ $\boxed{=37}$

SINGLE SOURCE SHORTEST PATH :- (Greedy)

↳ Dijikstra (Positive edges)

↳ Bellman ford (Negative edges)

Dijikstra algorithm:-

This algo. solves single source shortest path problem on a weighted directed graph. In this, all the edges are non-negative.

ALGORITHM:-

Dijikstra (G, w, s)

1) Initialise single source (G, W, s)

2) $S \leftarrow \phi$

3) $Q \leftarrow V[G]$

4) while $Q \neq \phi$

5) do $u \leftarrow$ extract min (Q)

6) $S \leftarrow S \cup \{u\}$

7) for each vertex $v \in Adj(u)$

8) do relax (u, v, w)

Initialise single source (G, s)

1) for each vertex $v \in V[G]$

2) do $d[v] \leftarrow \infty$

3) $\pi[v] \leftarrow$ Nil

4) $d[s] \leftarrow 0$

~~5) do u ← ext~~

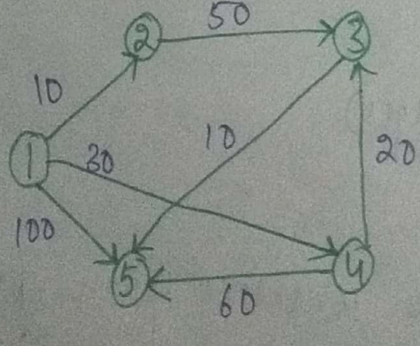
Relax (u, v, w)

1) if $d[v] > d[u] + w[u,v]$

2) then $d[v] \leftarrow d[u] + w[u,v]$

3) $\pi[v] \leftarrow u$

Ex →



Considering '1' as source vertex

Q =

0	∞	∞	∞	∞
1	2	3	4	5

Parent

Nil	Nil	Nil	Nil	Nil
1	2	3	4	5

1) Extract (1)

Adj (1) ∈ 2, 4, 5

$d[2] > d[1] + d[1,2]$

10	∞	∞	∞
2	3	4	5

$\infty > 0+10$

$\infty > 10$

$d[4] > d[1] + d[1,4]$

10	∞	30	∞
2	3	4	5

$\infty > 0+30$

$d[4] = 30$

$d[5] > d[1] + d[1,5]$

10	∞	30	100
2	3	4	5

Nil	1	Nil	1	1
1	2	3	4	5

→ Parent

$\infty > 0+100$

$d[5] \leftarrow 100$

2) Extract (2)

Adj (2) ∈ 3

$d[3] > d[2] + d[2,3]$

60	30	100
3	4	5

Nil	1	2	1	1
1	2	3	4	5

$\infty > 10+50$

$\infty > 60$

$d[3] \leftarrow 60$

3) Extract (4)

Adj (4) ∈ 3, 5

$d[3] > d[4] + w[4,3]$

$60 > 30+20$

$60 > 50$

50	100
3	5

$d[3] \leftarrow 50$

$d[5] > d[4] + w[4,5]$

$100 > 30+60$

$100 > 90$

50	90
3	5

Nil	1	4	1	4
1	2	3	4	5

$d[5] \leftarrow 90$

4) Extract(3)

Adj(3) ← 5

d[5] > d[3] + w[3,5]

90 > 50 + 10

90 > 60

d[5] ← 60

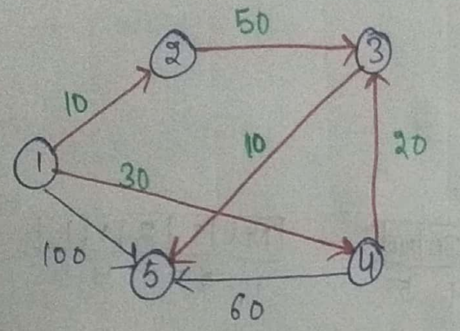
	60				
	5				

Nil	1	4	1	3
1	2	3	4	5

5) Extract(5)

Adj(5) ← Nil

Final tree :-



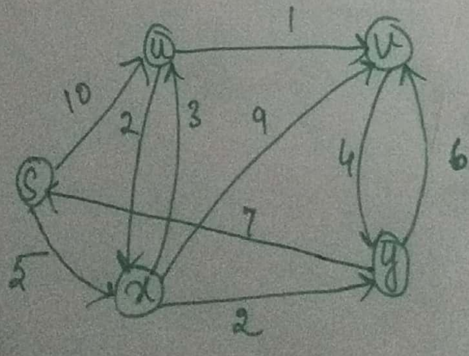
1-2 = 10

1-3 = 1-4-3 = 30+20 = 50

1-5 = 1-4-3-5 = 30+20+10 = 60

1-4 = 30

Q:-



Source vertex = s

Q =

0	∞	∞	∞	∞
s	u	v	x	y

Nil	Nil	Nil	Nil	Nil
s	u	v	x	y

Scanned by CamScanner

Extract(s)

Adj(s) ∈ u, x

$d[u] > d[s] + d[s,u]$

$\infty > 0+10$

$\infty > 10$

$d[u] \leftarrow 10$

10	∞	∞	∞
u	v	x	y

$d[x] > d[s] + d[s,x]$

$\infty > 0+5$

$\infty > 5$

$d[x] \leftarrow 5$

10	∞	5	∞
u	v	x	y

Nil	S	Nil	S	Nil
s	u	v	x	y

Extract(x)

Adj(x) ∈ u, v, y

$d[u] > d[x] + w[x,u]$

$10 > 5+3$

$10 > 8$

$d[u] \leftarrow 8$

8	∞	∞
u	v	y

$d[v] > d[x] + w[x,v]$

$\infty > 5+9$

$\infty > 14$

$d[v] \leftarrow 14$

8	14	∞
u	v	y

$d[y] > d[x] + w[x,y]$

$\infty > 5+2$

$\infty > 7$

$d[y] \leftarrow 7$

8	14	7
u	v	y

Nil	x	x	s	x
s	u	v	x	y

Extract(y)

Adj(y) ∈ v, s

$d[v] > d[y] + w[y,v]$

$14 > 7+6$

Scanned by CamScanner

14>13

d[v]←13

u	v
8	13

s	u	v	x	y
Nil	x	y	s	x

Extract (u)

Adj (u) ∈ x, v

$d[v] > d[u] + w[u,v]$

$13 > 8+1$

$13 > 9$

d[v]←9

v
9

s	u	v	x	y
Nil	x	u	s	x

Extract (v)

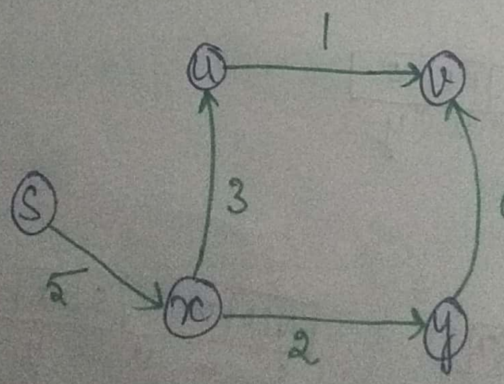
Adj(v) → nil

s-x = 5

s-u = 8

s-y = s-x-y = 7

s-v = s-x-u-v = 9



Greedy	Dynamic programming
• It is a top-down approach.	• It is a bottom up approach.
• Only one decision sequence is generated.	• Many decision sequences are generated.
• It makes whatever choice seems best at the moments, then solve the subproblem.	• Depends upon the further choices or solutions of a subproblem.
• Subproblems are independent.	• All subproblems are dependent on each other.

0/1 KNAPSACK PROBLEM :-

$n = 5$, $W = 60$

$(w_1, w_2, w_3, w_4, w_5) = (5, 10, 20, 30, 40)$

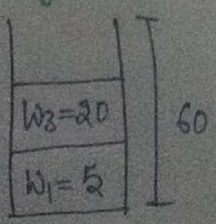
$(P_1, P_2, P_3, P_4, P_5) = \{30, 20, 100, 90, 160\}$

$\frac{P_1}{w_1} = 6$, $\frac{P_2}{w_2} = \frac{20}{10} = 2$, $\frac{P_3}{w_3} = \frac{100}{20} = 5$, $\frac{P_4}{w_4} = \frac{90}{30} = 3$

$\frac{P_5}{w_5} = \frac{160}{40} = 4$

$$\frac{P_1}{w_1} > \frac{P_3}{w_3} > \frac{P_5}{w_5} > \frac{P_4}{w_4} > \frac{P_2}{w_2}$$

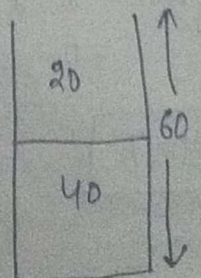
Greedy :-



$= \{1, 0, 1, 0, 0\}$

Max. profit $= 1 \times 30 + 1 \times 100$
$= 130$

Dynamic :-



Max. profit $= 160 + 100$
$= 260$

0/1 KNAPSACK (Dynamic programming)

$n = 4$, $W = 5$

$w_1, w_2, w_3, w_4 = \{2, 3, 4, 5\}$

$v_1, v_2, v_3, v_4 = \{3, 4, 5, 6\}$

i \ w	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	3	3	3	3
2	0	0	3	4	4	3+4=7
3	0	0	3	4	5	7
4	0	0	3	4	5	$\boxed{7}$

→ Profit matrix

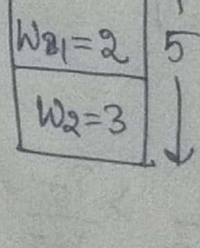
Max. profit

same values,
∴ discard item 4

similarly, discard item 3

3 & 7 are different, ∴ include item 2

Rem. weight $= 5 - 3$
$= 2$



then, move to weight 2 in profit matrix

0 & 3 are different, ∴ include item 1

Rem. weight $= 2 - 2$
$= 0$

$\boxed{\text{Max. profit} = 3 + 4 = 7}$

Scanned by CamScanner

Q:- $n=7$, $m=15$

$(w_1, w_2 --- w_7) = (2, 3, 5, 7, 1, 4, 1)$

$(P_1, P_2 \longrightarrow P_7) = (10, 5, 15, 7, 6, 18, 3)$

i \ w	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0	0	10	10	10	10	10	10	10	10	10	10	10	10	10	10	
2	0	0	10	10	10	15	15	15	15	15	15	15	15	15	15	15	
3	0	0	10	10	10	15	15	25	25	25	30	30	30	30	30	30	
4	0	0	10	10	10	15	15	25	25	25	30	30	30	30	32	32	→ diff
5	0	6	10	16	16	16	21	25	31	31	31	36	36	36	36	38	→ diff.
6	0	6	10	16	18	24	28	34	34	34	49	43	49	49	49	54	→ same
7	0	6	10	16	19	24	28	34	37	37	39	43	49	52	52	54	

Max. profit

→ Discard item 7

→ Include " 6

15−4=11

→ Keep 5

11−1=10

→ Discard 4

→ Keep 3

10−5=5

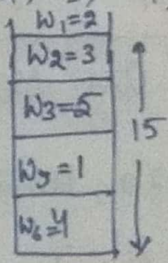
→ Keep 2

5−3=2

→ Keep 1

2−2=0

$\{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$



$\boxed{\text{Max. profit} = 54}$

ALGO :- (0/1 Knapsack using DP)

1) for w = 0 to W
2) B[0, w] = 0
3) for i = 1 to n
4) B[i, n] = 0
5) for i = 1 to n
6) for w = 0 to W
7) if w_i <= w {
8) if b_i + B[i-1, w-w_i] > B[i-1, w] {
9) B[i, w] ← b_i + B[i-1, w-w_i] }
10) else {
11) B[i, w] = B[i-1, w] } }
}

MATRIX CHAIN MULTIPLICATION :-

→ It is used in compiler design in code optimization problem.

$M_1 = 10*20$, $M_2 = 20*50$, $M_3 = 50*1$, $M_4 = 1*100$

$M = M_1 \cdot (M_2 \cdot (M_3 \cdot M_4))$

50*100 = 50*100*1 = 5000

20*100 = 20*50*100 = 100000

10*100 = 10*20*100 = 20000

$\boxed{\text{Cost} = 125000}$

$M = (M_1 \cdot (M_2 \cdot M_3)) \cdot M_4$

20*1 = 20*50*1 = 1000

10*1 = 10*1*20 = 200

10*100 = 10*1*100 = 1000

$\boxed{\text{Cost} = 2200}$