DAA (IMP) NOTES

Date : Kausar.

$$\sum_{i=1}^{K} i \cdot 2^i$$

$= Sk = 2 + 2 \cdot 2^2 + 3 \cdot 2^3 + 4 \cdot 2^4 + \dots + k \cdot 2^k$ - (1)

multiply eq (1) by 2

$2Sk = 2^2 + 2 \cdot 2^3 + 3 \cdot 2^4 + \dots + k \cdot 2^{k+1}$ - (2)

(1) - (2)

$Sk - 2Sk = (2 + 2 \cdot 2^2 + 3 \cdot 2^3 + 4 \cdot 2^4 + \dots + k \cdot 2^k) - (2^2 + 2 \cdot 2^3 + 3 \cdot 2^4 + \dots + k \cdot 2^{k+1})$

$(1-2)S = 2 + 2^2 + 2^3 + 2^4 + \dots + 2^k - k \cdot 2^{k+1}$

$-Sk = 2(\underline{1 + 2 + 2^2 + 2^3 + \dots + 2^{(k-1)}}) - k \cdot 2^{k+1}$

$Sk = \frac{a(1-r^n)}{1-r}$

$a = 1, r = 2$

$-Sk = 2\left[\frac{1(1-2^k)}{(1-2)}\right] - k \cdot 2^{k+1}$

$-Sk = 2\left[\frac{1-2^k}{-1}\right] - k \cdot 2^{k-1}$

$Sk = 2(1-2^k) + k \cdot 2^{k-1}$

Page No.

Date :

$$\sum_{i=1}^{k-1} i \cdot a^i$$

$\Rightarrow$ $S_k = a + 2a^2 + 3a^3 + 4a^4 + \dots + (k-1)a^{(k-1)}$ — (1)

multiply eq (1) by a

$aS_k = a^2 + 2a^3 + 3a^4 + \dots + (k-1)a^k$ — (2)

(1) - (2)

$S_k - aS_k = (a + 2a^2 + 3a^3 + \dots + (k-1)a^{(k-1)}) - (a^2 + 2a^3 + 3a^4 + \dots + (k-1)a^k)$

$(1-a)S_k = a + a^2 + a^3 + \dots + a^{(k-1)} - (k-1)a^k$

$= a + a^2 + a^3 + \dots + a^{k-1} - ka^k + a^k$

$= \underline{a + a^2 + a^3 + a^4 + \dots + a^{k-1} + a^k} - ka^k$

$= a(1 + a + a^2 + a^3 + \dots + a^{k-1}) - ka^k$

$(1-a)S_k = a\left[\frac{1(1-a^k)}{(1-a)}\right] - ka^k$

$= \frac{a(1-a^k)}{(1-a)} - ka^k$

$S_k = \frac{a(1-a^k)}{(1-a)^2} - \frac{ka^k}{(1-a)}$

Date :

Recurrance Relation

1) Characteristics equation.

i) Homogeneous Recurrance.

i) $t_n = \begin{cases} n & \text{if } n=0 \text{ or } n=1 \\ 5t_{n-1} - 6t_{n-2} & \text{otherwise} \end{cases}$

$t_n = 5t_{n-1} - 6t_{n-2}$

$t_n - 5t_{n-1} + 6t_{n-2} = 0$

put $t_n = x^n$

$x^n - 5x^{n-1} + 6x^{n-2} = 0$

$n = 2$

$x^2 - 5x + 6 = 0$

$r_1 = 3 , r_2 = 2$

$t_n = c_1 r_1^n + c_2 r_2^n$

$t_n = c_1 3^n + c_2 2^n$ — (1)

when $n=0$ in eq (1)

$t_0 = c_1 + c_2$ — $\cancel{(2)}$

$\cancel{t_0}$ $c_1 + c_2 = 0$ — (2)

$t_1 = 3c_1 + 2c_2$

$3c_1 + 2c_2 = 1$ — (3)

Date :

1. Charateristics Method

1) Homogeneous - $t_n = \sum_{i=1}^{k} c_i r_i^n$ $= 0$

2) Inhomogeneous - Roots

1) Real and distinct

$t_n = c_1 r_1^n + c_2 r_2^n + c_3 r_3^n.$ $\neq 0$

2) Real and similar roots

$t_n = c_1 n^0 r_1^n + c_2 n^1 r_2^n + c_3 n^2 r_3^n + c_4 n^3 r_4^n.$

3) Logarithmic - Same as Inhomogeneous

3. Master Method - $\boxed{T(n) = aT(n/b) + f(n)}$

Case 1 :- If $f(n) = \Theta(n^{\log_b a - \epsilon})$ where if $\log_b^a > f(n)$

then $T(n) = \Theta(n^{\log_b^a})$

Case 2 :- If $f(n) = \Theta(n^{\log_b^a})$ $\log_b^a = f(n)$

then $T(n) = \Theta(n^{\log_b^a} \cdot \lg n)$

Case 3 :- If $f(n) = \Theta(n^{\log_b a + \epsilon})$ where $\epsilon > 1$ $\log_b^a < f(n)$

Then $T(n) = \Theta((f(n)))$

2. Recursion Tree Method

$T(n) = 2T(n/2) + \Theta(n) \longrightarrow$ cost of original recurrance

Original recurrance

Sub recurrance (Sub-problem)

Size of Sub-recurrance

No. of Sub-recurrance

Date :

Formula.

1) $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

2) $\sum_{i=2}^{n} i = \frac{n(n+1)}{2} - 1$

3) $\sum_{i=3}^{n} i = \frac{n(n+1)}{2} - 3$

4) $\sum_{i=4}^{n} i = \frac{n(n+1)}{2} - 6$

1) $\sum_{i=1}^{n} (1) = 1 \times n$

2) $\sum_{i=2}^{n} (1) = 1 \times (n-1)$

3) $\sum_{i=3}^{n} (1) = 1 \times (n-2)$

4) $\sum_{i=4}^{n} (1) = 1 \times (n-3)$

Insertion Sort

Algorithm

c_1	n	for $j \rightarrow 2$ to length(A)
c_2	$n-1$	key = A(j)
c_3	$n-1$	i = j-1
c_4	$\sum_{j=2}^{n} t_j$	while i>0 ~~and~~ & A(i)>key
c_5	$\sum_{j=2}^{n} t_j - 1$	A(i+1) ← A(i)
c_6	$\sum_{j=2}^{n} t_j - 1$	i ← i-1
c_7	$n-1$	A(i+1) ← key

For $j \rightarrow 2$ to length(A)
key = A(j)
i = j-1
while i>0 & A(i)>key
~~A(j)~~ =
A(i+1) ← A(i)
i ← i-1
A(i+1) ← key

Date :

A(j) A(1) A(j) A(3) A(4) A5

A = < 8 , 6 , 5 , 4 , 2 >

For j = 2 to 5

j = 2

key = 6

i = 1

while 1 > 0 & 8 > 6 (T)

A(2) = 8

i = 0

while 0 > 0 & A(0) > 6 (F)

A(1) = 6

6	8	5	4	2

For j = 3

~~j~~ key = 5

i = 2

while 2 > 0 & $\overbrace{8 > 5}^{A(i) > key}$ (T)

A(3) = 8

i = 1

while 1 > 0 & 6 > 5 (T)

A(2) = 6

i = 0

while 0 > 0 & A(0) > 5 (F)

A(1) = 5

5	6	8	4	2

For j = 4

key = 4

i = 3

while 3 > 0 & 8 > 4 (T)

A(4) = 8

i = 2

while 2 > 0 & 6 > 4 (T)

Date :

A(3) = 6

i = 1

while 1>0 & 5>4 (T)

A(2) = 5

i = 0

while 0>0 & A(0)>4 (F)

A(1) = 4

4	5	6	8	2

For j = 5

key = 2

i = 4

while 4>0 & 8>2 (T)

A(5) = 8

i = 3

while 3>0 & 6>2 (T)

A(4) = 6

i = 2

while 2>0 & 5>2 (T)

A(3) = 5

i = 1

while 1>0 & 4>2 (T)

A(2) = 4

i = 0

while 0>0 & A(0)>2 (F)

A(1) = 2

2	4	5	6	8

Time complexity = $\Theta(n^2)$

= $O(n^2)$

Date :

- Time complexity

$$T(n) = aT(n/b) + \theta(D) + \theta(c)$$

division — quotient

$$= 1T(n/2) + 1 + 1$$
$$= T(n/2) + 1$$
$$Tn = T/n/2$$

~~Time complexity~~

* Divide and Conquer Strategy

Time complexity

$$f(n) = n^d$$
$$n^{\log_b^a} = n^{\log_2^1}$$
$$= n^{\lg n}$$
$$= n^0$$
$$T(n) = \theta(n^{\log_b^a} \cdot \lg n)$$
$$= \theta(n^0 \cdot \lg n)$$
$$T(n) = \theta(\lg n)$$

1) Binary Search

```
Binary-search (A, x, l, r)
    if l == r
        return l
    else  m = [(l+r)/2]
        if x ≤ A(m) then
            return Binary-search (A, x, l, m)
        else
            return Binary-search (A, x, m+1, r)
```

$$T(n) = \theta(n^{\log_b^a} \cdot \lg n)$$
$$= \theta(n^0 \cdot \lg n)$$
$$T(n) = \theta(\lg n)$$

A(0) A(1) A(2) A(3) A(4) A(5) A(6) A(7) A(8) A(9)

A(10) = < 7, 9, 15, 18, 21, 26, 31, 38, 41, 55 >

x = 38

l = 0 , r = 9

Binary-search (~~10~~ A, 38, 0, 9)

m = [[(9+0)/2] = [(9)/2] = 4.5 = 4

if 38 ≤ A(4)

38 ≤ 21 (F)

return Binary-search (A, x, 5, 9)

26	31	38	41	55

Binary-search (A, 38, 5, 9)

m = [14/2] = 7

if 38 ≤ A(7)

38 ≤ 38) T (A, x, 5, 7)

26	31	38

Binary-search (A, 38, 5, 7)

m = 12/2 = 6

38 ≤ 31

return Binary-search (A, x, 7 7)

Binary search (A, 38, 7, 7)

if (7 = 7)

return (7)

Page No.

2. Strassen's Matrix Multiplication

Formulaes.

$P_1 = A_{11} \cdot B_{12} - A_{11} \cdot B_{22}$

$P_2 = A_{11} \cdot B_{22} + A_{12} \cdot B_{22}$

$P_3 = A_{21} \cdot B_{11} + A_{22} \cdot B_{11}$

$P_4 = A_{22} \cdot B_{21} - A_{22} \cdot B_{21}$

$P_5 = (A_{11} + A_{22})(B_{11} + B_{22})$

$P_6 = (A_{12} - A_{22})(B_{21} + B_{22})$

$P_7 = (A_{11} - A_{21})(B_{11} + B_{12})$

$C_{11} = P_5 + P_4 - P_2 + P_6$

$C_{12} = P_1 + P_2$

$C_{13} = P_3 + P_4$

$C_{14} = P_5 + P_1 - P_3 - P_7$

$P_1 = A_{11} \cdot B_{12} - A_{11} \cdot B_{22}$

$P_2 = A_{11} \cdot B_{22} + A_{12} \cdot B_{22}$

$P_3 = A_{21} \cdot B_{11} + A_{22} \cdot B_{11}$

$P_4 = A_{22} \cdot B_{21} - A_{22} \cdot B_{11}$

$P_5 = (A_{11} + A_{22})(B_{11} + B_{22})$

$P_6 = (A_{12} - A_{22})(B_{21} + B_{22})$

$P_7 = (A_{11} - A_{21})(B_{11} + B_{12}$

$C_{11} = P_5 + P_4 - P_2 + P_6$

$C_{12} = P_1 + P_2$

$C_{13} = P_3 + P_4$

$C_{14} = P_5 + P_1 - P_3 - P_7$

3. Merge Sort

Merge_sort (A,p,r)
 if $p < r$
 ~~mergesort~~
 $q = \lfloor (p+r)/2 \rfloor$
 merge_sort (A,p,q)
 merge_sort (A,q+1,r)
 merge_sort (A,p,q,r)

merge(A,p,q,r)
 $n_1 = q-p+1$
 $n_2 = r-q$
 for $i = 1$ to n_1
 $L(i) = A(p+i-1)$

merge sort (A,p,r)
 if $p < r$.
 $q = \lfloor (p+r)/2 \rfloor$
 merge sort (A,p,q)
 merge_sort (A,q+1,r)
 Merge-sort (A,p,q,r)

merge sort (A,p,q,r)
 $n_1 = q-p+1$
 $n_2 = r-q$
 for $i = 1$ to n_1
 $L(i) = A(p+i-1)$

Date :

```
For j = 1 to n2
    R(j) = A(q-j)
L(n1+1) = ∞
R(n2+1) = ∞
    i = 1 , j = 1
for k = p to r
    if L(i) ≤ R(j)
        A(k) = L(i)
        i = i+1
    else
        A(k) = R(j)
        i = j+1
```

```
For j = 1 to n2
    R(j) = A(q-j)
L(n1+1) = ∞
R(n2+1) = ∞
    i = 1 , j = 1
for k = p to r
    if L(i) ≤ R(j)
        A(k) = L(i)
        i = i+1
    else
        A(k) = R(j)
        i = j+1
```

1. Quick Sort

```
1) Quick_sort (A, p, r)
if p<r
    then  q = partition (A, p, r)
          Quicksort (A, p, q-1)
          Quicksort (A, q+1, r)

2) partition (A, p, r)
    x = A(r)
    i = p-1
    for j = p to r-1
        if A(j) ≤ x
            then  i = i+1
                  A(i) <--> A(j)
    A(i+1) <--> A(r)
    return i+1
```

```
1) Quicksort (A, p, r)
if p<r
    then  q = partition (A, p, r)
          Quicksort (A, p, q-1)
          Quicksort (A, q+1, r)

2) partition (A, p, r)
    x = A(r)
    i = p-1
    for j = p to r-1
        if A(j) ≤ x
            then  i = i+1
                  A(i) <--> A(j)
    A(i+1) <--> A(r)
    return i+1
```

★ Greedy Approach

1) Fractional knapsack Problem

⇒ i) Decreasing order of their profits
ii) Increasing order of their weights
iii) Decreasing order of their Pi/wi ratios

⇒ Job No. weight profit Remaining capacity of knapsack

2) Travelling Saleaman Problem

	a	b	c	d
a	0	5	$\boxed{2}$	3
b	2	0	4	[illegible]
c	6	$\boxed{3}$	0	5
d	$\boxed{2}$	1	6	0

Shortest path

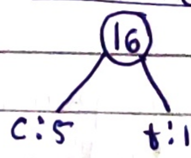
$a \xrightarrow{2} c \xrightarrow{1} d \xrightarrow{3} b \xrightarrow{2} a$

Minimus cost = 2+1+3+2
= 8.

3) Huffman code

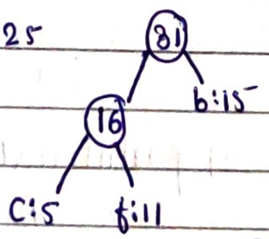
$\boxed{c:5 \quad f:11}$ b:15 a:20 d:25 e:35

1st arranging in ascending order

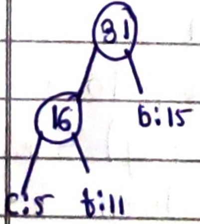


b:15 a:20 d:25 e:35

a:20 d:25

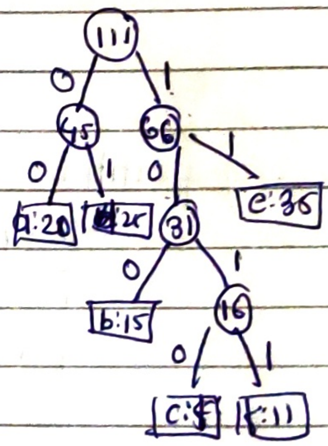


e:35



e:35

a:20 = 00
a:15 : 100
c:5 : 1010
d :25 : 01
e =35 = 11
f:11 = 1011



at the end.

Date :

4) Minimum spanning tree.

1) kruskal algo.

⇒ MST_kruskal (G, ω)

A ← ∅

for each vertex v ∈ V(G)

make_set (v)

for each ~~vertex~~ edge (u, v) ∈ E(G)

if find_set (u) ≠ find_set (v)

A = A + ω (u, v)

return A.

eg ⇒ Create Table

Edges in ascending order of their weights	weight of edge	Include/Exclude

2) Prim's algo

⇒ MST_prims(G, ω, r)

for each u ∈ V(G)

key(u) = ∞

π(u) = NIL

key (r) = 0

Q ← V(G)

while Q ≠ ∅

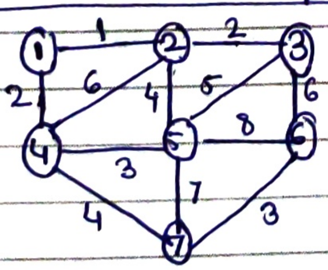
u ← extract_min (Q)

for each v ∈ adj(u)

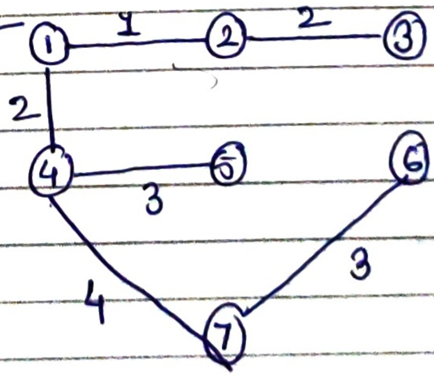
π(v) = u

key (v) = ω(u, v)

Q →	1	2	3	4	5	6	7
key →	0	∞	∞	∞	∞	∞	∞
	–	1	∞	2	∞	∞	∞
		–	2	2	4	∞	∞
		–	–	2	4	6	∞
		–	–	–	3	6	4
		–	–	–	–	6	4
						3	–
						–	



final



Date :

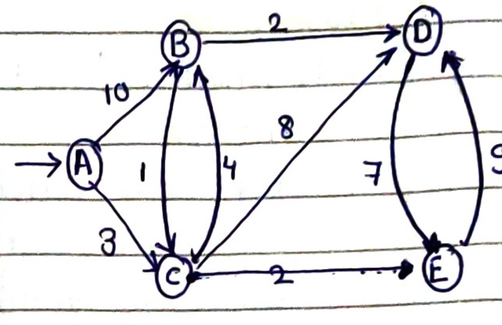
5) Single - Source Shortest path

Dijkstra's algo.

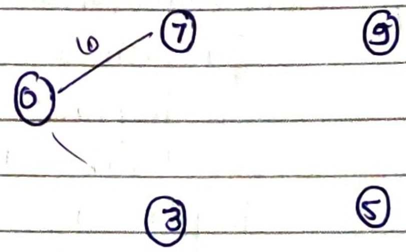
Dijkstra(G, w, s)
    initialize-single-source (G, s) — ①
    $S = \phi$
    $Q = V(G)$
    while ($Q \neq \phi$)
        u = Extract_min(Q)
        $S = S \cup \{u\}$
        for each vertex $v \in adj(u)$
            RELAX (u, v, w) — ②

2) Initialize_Single_source (G, s)
    for each vertex $v \in V(G)$.
        $v.d = \infty$
        $v.\pi$ = NIL
    $s.d = 0$

3) RELAX (u, v, w)
    if $v.d > u.d + w(u,v)$
        $v.d = u.d + w(u,v)$
        $v.\pi = u$.



Q →	A	B	C	D	E
	0	∞	∞	∞	∞
		10	$\boxed{3}$	∞	∞
		7	-	11	$\boxed{5}$
		$\boxed{7}$	-	11	-
		-	-	$\boxed{9}$	-
		-	-	-	-



Shortest path (A, B) = $d(A,C) + d(C,B)$
$= 3 + 4 = 7$

Date :

7) Job-sheduling with deadline

$i \rightarrow$	1	2	3	4	5
$p_i \rightarrow$	25	15	20	45	13
$d_i \rightarrow$	3	1	1	2	3

Max dedline 3

Input sequence   Sequence for processing   profit
(1,2,3)

* Dynamic programming

1) Longest common subsequence (LCS)   X = POLYNOMIAL   Y = EXPONENTIAL

Rule 1 :

	x_i	B
y_j	0	0
B	0	↖ 1

$\boxed{\nwarrow}$ point   Adding one to the digonal point if element are same i.e. B ↔ B.

Rule 2 :

	x_i	B
y_j	0	[illegible]
A	0	↑ [illegible]

$\boxed{\uparrow}$ point   element are different ~~and above~~ is. ~~less than or equal to~~

Condi.   A<B or equal to A=B

Rule 3 :

	x_i	B
y_j	0	0
A	1	← 1

$\boxed{\leftarrow}$ point   element are different and B<A i.e. ~~1<0~~
0<1

Date :

2) Floyd - Warshall algo.

$D(0) = \begin{bmatrix} 0 & 4 & 7 \\ 1 & 0 & 2 \\ 6 & \infty & 0 \end{bmatrix}$ $\quad \Pi(0) = \begin{bmatrix} N & 1 & 1 \\ 2 & N & 2 \\ 3 & N & N \end{bmatrix}$

$D(1) = \begin{bmatrix} 0 & 4 & 7 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$ $\quad \Pi(1) = \begin{bmatrix} N & 1 & 1 \\ 2 & N & 2 \\ 3 & 3 & N \end{bmatrix}$

$D(2) = \begin{bmatrix} 0 & 4 & 6 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$ $\quad \Pi(2) = \begin{bmatrix} N & 1 & 2 \\ 2 & N & 2 \\ 3 & 3 & N \end{bmatrix}$

$D(3) = \begin{bmatrix} 0 & 4 & 6 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$ $\quad \Pi(3) = \begin{bmatrix} N & 1 & 2 \\ 2 & N & 2 \\ 3 & 3 & N \end{bmatrix}$

3) 0/1 knapsack

$m[i, w] = \max\{m[i-1, w]\}, m[i-1, w-w[i]] + P_i$

weight = {3, 4, 6, 5}

profit = {2, 3, 1, 4}

w -- 8

$n = 4$

Pi	wi
2	3
3	4
4	5
1	6

wi →

	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	2	2	2	2	2
2	0	0	0	2	3	3	3	5	5
3	0	0	0	2	3	4	4	5	6
4	0	0	0	2	3	4	4	5	$\boxed{6}$

max(4+0, 3)

max(6+0, 4)

$x_1 \quad x_2 \quad x_3 \quad x_4$

1 , 0 , 0 , 1