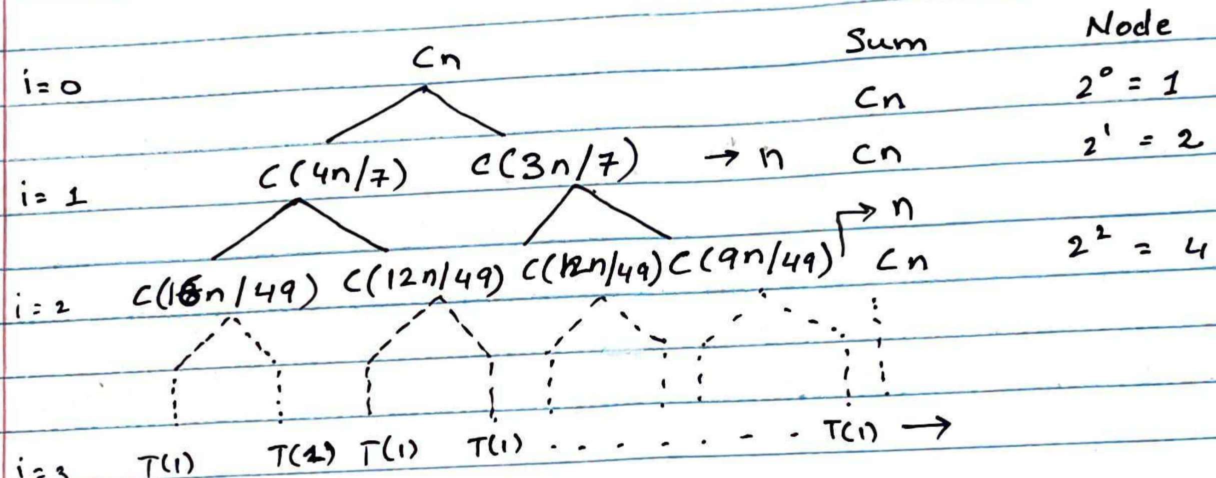
Algorithm Design - Assignment-3

Name : Pratyusha Reddy Thamminen
ID : 101707053

1. A recursion tree for the recurrence of
$T(n) = T(4n/7) + T(3n/7) + O(n)$

$O(n) = Cn$ where $C \geq 1$



$$\left(\frac{3}{7}\right)^{t_1} \times n = 1$$

$$n = (7/3)^{t_1}$$

$$\boxed{\therefore\ t_1 = \log_{7/3} n} \longrightarrow \text{I}$$

$\therefore$ The Sum at each level is
$n + n + n + \ldots\ldots + \log(n)$
$\boxed{n \log n}$

$\therefore$ Complexity is $O(n \log n)$

2. a. $T(n) = 3\,T(n/2) + n^4 + 300$

According to the master's theorem

$$\left.\begin{array}{l} T(n) = aT(n/b) + f(n) \\ \quad \& \quad f(n) = \Theta(n^d) \\ \quad \& \quad T(1) = c \end{array}\right\}$$

So $\Rightarrow$ $a = 3$ $\quad b = 2$ $\quad d = 4.$

where $a \geq$, $b \geq 2$, and $d \geq 0$, $c > 0$

$$\text{By } T(n) = \begin{cases} \Theta(n^d) \text{ if } \log_b a < d \text{ (or) } a < b^d \\ \Theta(n^d \log n) \text{ if } \log_b a = d \text{ (or) } a = b^d \\ \Theta(n^{\log_b a}) \text{ if } \log_b a > d \text{ (or) } a > b^d \end{cases}$$

$\therefore$ Now

$3 < 2^4$

Hence $T(n) \in \Theta(n^d)$

$\therefore$ $T(n) = \Theta(n^4)$ is the complexity.

b. $T(n) = T(n/2) + (3/5)n^2$

According to master's theorem.

$$\left.\begin{aligned} T(n) &= aT(n/b) + f(n) \\ &\& \ f(n) = \Theta(n^d) \\ &\& \ T(1) = c \end{aligned}\right\}$$

$\therefore$ $a = 1$ $b = 2$ $d = 2$.

From the above expression.

$1 < 2^2$

$\therefore$ $T(n) = \Theta(nd)$

$T(n) = \Theta(n^2)$ is the complexity.

c. $T(n) = 6T(n/2) + 6n^2$

According to master's theorem.

$\therefore$ $a = 6$ $b = 2$ $d = 2$.

FROM THE ABOVE EXPRESSION

$6 > 2^2$ $\boxed{2^2 = 4}$

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

$T(n) = \Theta(n^{\log_2 6})$ is the complexity.

The complexity is $T(n) = \Theta(n^{\log_2 6})$

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

Partitioning (A, B, d)

1. $i < p-1$
2. for $i \leftarrow p$ to $r-1$
3. if $A[i] \leq A|r$
4. $i \leftarrow i+1$
5. Swap $A[i]$ & $A[j]$
6. Swap $A[i+1]$ & $A[r]$

Now,

Step 1:

```
c
4 6 3 1 2 5    c = current item
i j        p   p = Pivot
```

$j = 0$ to $6-1$ $\quad i = -1 \quad c < p$

4 is less than 5, so we swap pointers & move the pointers

Step 2:

```
  c
4 6 3 1 2 5
i j       p
```

$c > p$, we move c

Step 3:

```
    c
4 6 3 1 2 5
  i j     p
```

$c < p$, we swap $A[i]$ & $A[j]$

Step 4:

```
    c
4 3 6 1 2 5
    i j   p
```

$c < p$, we again swap $A[i]$ & $A[j]$

Step 5:

```
      c p
4 3 1 6 2 5
    i j
```

$c < p$; we swap $A[i]$ & $A[j]$

Step 6:

$$\begin{array}{cccccc} & & & & & c \\ 4 & 3 & 1 & 2 & 6 & 5 \\ & & & & & p \\ & & & i & j & \end{array}$$

$c < p$; we swap $A[i]$ & $A[j]$

Step 7:

$$\begin{array}{|c|c|c|c|c|c|} \hline 4 & 3 & 1 & 2 & 5 & 6 \\ \hline \end{array}$$

p

Let's do partition based on pivot position

step(i) :-

$$\begin{array}{c|c|c|c|c|} & 4 & 3 & 1 & 2 \\ i & j & & & p \end{array}$$

step(ii) $c \ngtr p$

$$\begin{array}{c|c|c|c|} & c & & \\ 4 & 3 & 1 & 2 \\ i & j & & p \end{array}$$

step (iii)

$$\begin{array}{|c|c|c|c|} & & c & p \\ 4 & 3 & 1 & 2 \\ i & j & & \end{array}$$

$c < p$, swap $A[i]$ & $A[j]$

step (iv)

$$\begin{array}{|c|c|c|c|} & & & c\ p \\ 1 & 3 & 4 & 2 \\ & i & & j \end{array}$$

$c = p$ swap $A[i]$ & $A[j]$

Step (v) $\boxed{1 \mid 2 \mid 3 \mid 4}$

$\therefore$ Final array $\boxed{1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6}$

4.

Item	Value	weight
1	5	2
2	10	4
3	15	5
4	20	7
5	25	9

Given weight = 12.

$\boxed{W=12}$

a. Table that shows the knapsack dynamic programming for optimum solution.

Item	Value	weight	0	1	2	3	4	5	6	7	8	9	10	11	12
1	5	2	0	0	5	5	5	5	5	5	5	5	5	5	5
2	10	4	0	0	5	5	10	10	15	15	15	15	15	15	15
3	15	5	0	0	5	5	10	15	15	20	20	25	25	30	30
4	20	✓ $\boxed{7}$	0	0	5	5	10	15	15	20	20	25	25	30	35 ←
✗ 5	25	9	0	0	5	5	10	15	15	20	20	25	25	30	$\boxed{35}$ ←

Optimum value achieved for this problem is 35.

b.

And according to the optimum value, items are :- {3, 4}

The objects → {4, 3} Total = 35

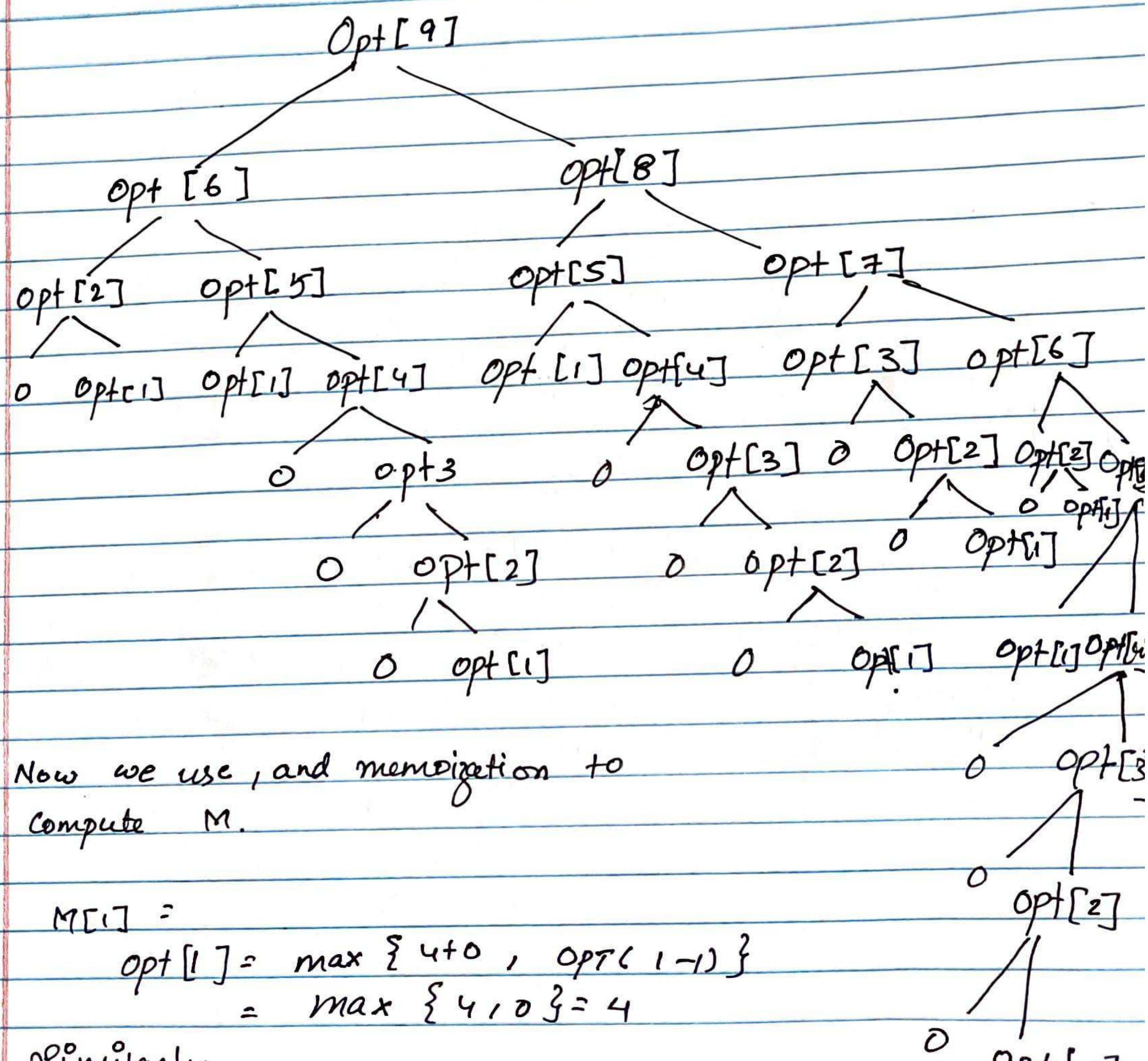
Value of 4 = 35
difference 15

3 2 15
difference

5. We have a job with their start time, end time and weight.

a) Now we do the top-down dynamic programming technique and make Recursion tree.

fibonacci Series.



Now we use, and memoization to Compute M.

M[1] =

$opt[1] = \max\{4+0,\ OPT(1-1)\}$

$= \max\{4, 0\} = 4$

Similarly.

$M[2] = opt[2] = \max\{5+0,\ opt(2-1)\}$

$= \max\{5, 4\}$

$= 5$

$M[3] = opt[3] = \max\{7+0,\ opt(3-1)\}$
$= \max\{7, 5\} = 7$

$M[4] = opt[4] = \max\{6+0,\ opt(4-1)\}$
$= \max\{6, 7\} = 7$

$M[5] = opt[5] = \max\{5+4,\ opt(5-1)\}$
$= \max\{9, 7\} = 9$

$M[6] = opt[6] = \max\{5+5,\ opt(6-1)\}$
$= \max\{10, 9\} = 10$

$M(7) = opt[7] = \max\{5+7,\ opt[7-1]\}$
$= \max\{12, 10\} = 12$

$M[8] = opt[8] = \max\{4+9,\ OPT[8-1]\}$
$= \max\{13, 12\} = 13$

$M[9] = OPT[9] = \max\{4+10,\ OPT[9-1]\}$
$= \max\{14, 13\} = 14$

Hence Now the table (M-table) is

0	1	2	3	4	5	6	7	8	9
M →	4	5	7	7	9	10	12	13	14

b. To find the ids issued, we need start and end time of each job.

(i) if we take $\{1, 5, 9\}$ a $\{1, 5, 8\}$
(ii) if we take $\{3, 7\}, \{3, 8\}, \{3, 9\}$
(iii) if we take $\{2, 6\}, \{2, 7\} \{2, 8\} \{2, 9\}$
(iv) if we take $\{2, 6, 9\}$

These all optimum scheduled Jobs.

$\therefore$ The id of the Schedule jobs from the table $\{9, 6, 2\}$ in order.

$\therefore$ The time is $\{9 \ 6 \ 2\}$

$$\underbrace{4 + 5 + 5} = 14$$

$\therefore$ The optimal Solution for opt (9) = 14 time.

6. $S_1 = \{ATMTCT\}$ 2 mismatches
$S_2 = \{ATCTMT\}$ 1 Gap.

a. Given mismatch $(\alpha) = 1$
Gap $(\delta) = 1$

	Col	A	T	M	T	C	T
Row	0	1	2	3	4	5	6
A	1	0	1	2	3	4	5
T	2	1	0	1	2	3	4
C	3	2	1	1	2	2	3
T	4	3	2	2	1	2	2
M	5	4	3	2	2	2	3
T	6	5	4	3	2	3	2

$\therefore$ Optimal Solution is "2"

6 b) Back tracting

		A	T	M	T	C	T
	0	1	2	3	4	5	6
A	1	0	1	2	3	4	5
T	2	1	0	1	2	3	4
C	3	2	1	1	2	2	3
T	4	3	2	2	1	2	2
M	5	4	3	2	2	2	3
T	6	5	4	3	2	3	2

A T M T C T

| | | | | |

A T C T M T

0 0 α 0 α 0

2α ; $\alpha = 1$

$2(1) = 2$ //