6|9|2023

DP-3

Knapsack problem: Given N objects with their values V_i, (profit/loss) & weight w_i. A bag is given with capacity W that can be used to carry some objects s.t.

total sum of object weights <= W &
sum of profit in the bag is maximized
loss minimized.

Fractional Knapsack. (objects can be divided)

Q Given N cakes with their happiness & weight. Find max total happiness that can be kept in a bag with capacity = W (cakes can be divided)

N = 5
W = 40

$$\begin{array}{lccccc} & 1 & 2 & 3 & 4 & 5 \\ h = [& 3 & 8 & 10 & 2 & 5] \\ w = [& 4 & 10 & 20 & 8 & 15] \end{array}$$

0.25 $5/15 \Rightarrow 0.33$

$8 \rightarrow 2$
$1 \rightarrow 2/8$

Rem cap	H
40	0
20 (40-20)	10
10 (20-10)	18
10-4(6)	21

$$\begin{array}{lccccc} & 1 & 2 & 3 & 4 & 5 \\ h = [& 3 & 8 & 10 & 2 & 5] \\ w = [& 4 & 10 & 20 & 8 & 15] \\ & \downarrow & \downarrow & \downarrow & \downarrow & \searrow \\ & 3/4 & 8/10 & 10/20 & 2/8 & 5/15 \\ & \Rightarrow 0.75 & : 0.8 & 0.5 & 0.25 & 0.33 \end{array}$$

$$\begin{array}{lccccc} & 1 & 2 & 3 & 4 & 5 \\ h = [& 3 & 8 & 10 & 2 & 5] \\ w = [& 4 & 10 & 20 & 8 & 15] \end{array}$$

Rem cap	H
40	0
(40-10)=30	8
(30-4)=26	11
(26-20) = 6	21
0	21+(6×0.33)
	⇒ 23(22...)

Greedy → w.r.t happiness of each part of cake.
$h[i]/wt[i]$

Soln ⇒ sort w.r.t $h[i]/wt[i]$ & select the cakes in descending order till the capacity is consumed.

$$\frac{x}{y} > \frac{a}{b} \Rightarrow \frac{xb}{yb} \quad \frac{ay}{yb}$$

$$\frac{0.25}{8} \qquad \frac{0.25}{6} \qquad 10$$

TC: O(N log N)
SC: O(1)/O(N)

0-1 Knapsack (object cannot be divided)

Q Given N toys with their happiness & weight. Find max total happiness that can be kept in a bag with capacity = W (toys cannot be divided)

N = 4

W = 7

$$\begin{array}{ccccc} & 1 & 2 & 3 & 4 \\ h = [& 4 & 1 & 5 & 7 \;] \\ w = [& 3 & 2 & 4 & 5 \;] \\ h/w & 1.33 & 0.5 & 1.25 & 1.4 \end{array}$$

W	H
7	0
(7-5)=2	7
(2-2)=0	8

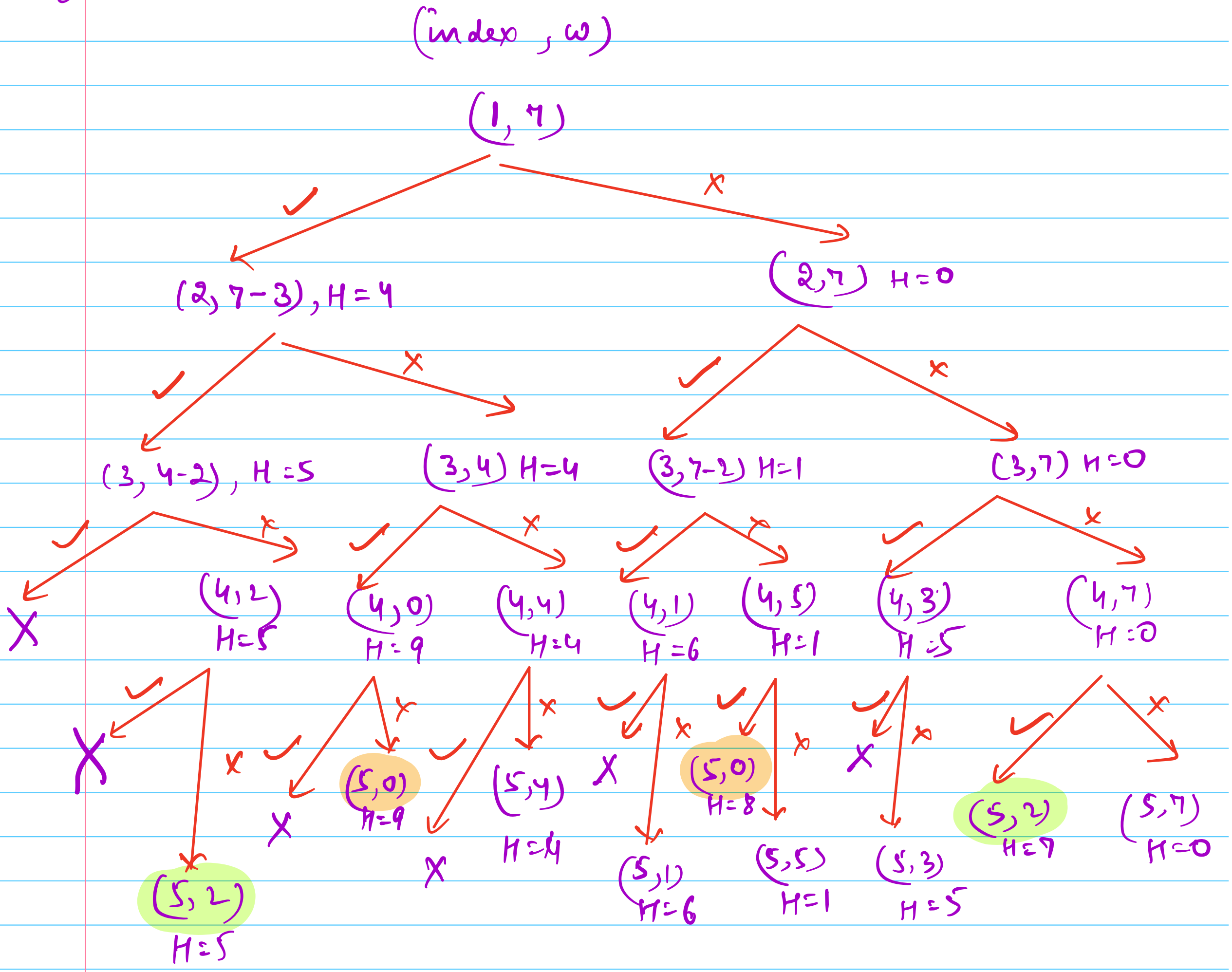
greedy ✗

$$\begin{array}{ccccc} & 1 & 2 & 3 & 4 \\ h = [& 4 & 1 & 5 & 7 \;] \\ w = [& 3 & 2 & 4 & 5 \;] \end{array}$$

9

Bruteforce → ∀ subsets of toys, s.t $\Sigma_{selected}\, w[i] <= W$, store the max happiness.

$$\begin{array}{lcccc} & 1 & 2 & 3 & 4 \\ h = [& 4 & 1 & 5 & 7\ ] \\ w = [& 3 & 2 & 4 & 5\ ] \end{array}$$



① optimal substructure
② overlapping subproblems. > DP

unique states → (1 - N) (0 - W)

⇒ $N * (W+1)$

= $O(NW)$

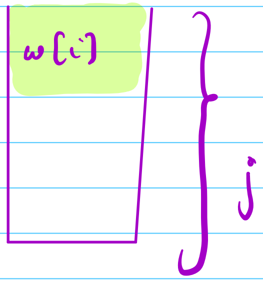
lot of states are repeating or invalid.

$dp[\overset{\text{object}}{\overset{\uparrow}{i}}][j] \rightarrow \text{capacity}.$

⇒ max happiness considering first i objects & capacity j

∀ j dp[0][j] = 0 (no toys)   ∀ i dp[i][0] = 0 (no capacity)

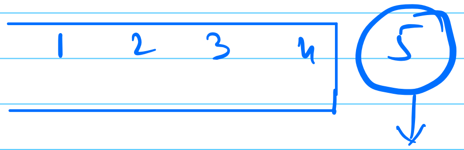
$$\begin{tikzcd}
dp[i][j] \arrow[r, "\times"] \arrow[dr, "\checkmark"'] & dp[i-1][j] \\
 & h[i] + dp[i-1][j-w[i]]
\end{tikzcd}$$



⑩   5th , H[5] = 10
w[5] = 6

10 − 6 = 4   (j − w[5]) =

dp[i-1][j-w[5]]



∀ i,j dp[i][j] = 0

```
for i → 1 to N {  // 1 based index
    for j → 1 to w {
        if (wt[i] > j)  dp[i][j] = dp[i-1][j]
        else {
            dp[i][j] = max ( dp[i-1][j] ,
                        h[i] + dp[i-1][j - wt[i]] );
        }
    }
```

}

return dp[N][W];

TC: O(N*W)

SC: O(N*W) → O(2W) ≃ O(W)

$$\begin{tikzcd} dp[i][j] \arrow[r, "\times"] \arrow[dr, "\checkmark"'] & dp[i-1][j] \\ & h[i] + dp[i-1][j-w[i]] \end{tikzcd}$$

$$\begin{array}{rcccc} & 1 & 2 & 3 & 4 \\ h = [& 4 & 1 & 5 & 7\ ] \\ w = [& 3 & 2 & 4 & 5\ ] \end{array}$$

4, 5 + dp[3-1][4-4]

N = 4

W = 7

j →

i ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	4	4	4	4	4
2	0	0	1	4	4	5	5	5
3	0	0	1	4	5	5	6	9
4	0	0	1	4	5	7	7	9 (Ans)

5, dp[2][7-4] + 5 , 6, 7 + dp[3][1]

5, dp[3][0] + 7 9, 7 + dp[3][2]

Meet at 8:55 am IST

Unbounded Knapsack / 0-N Knapsack

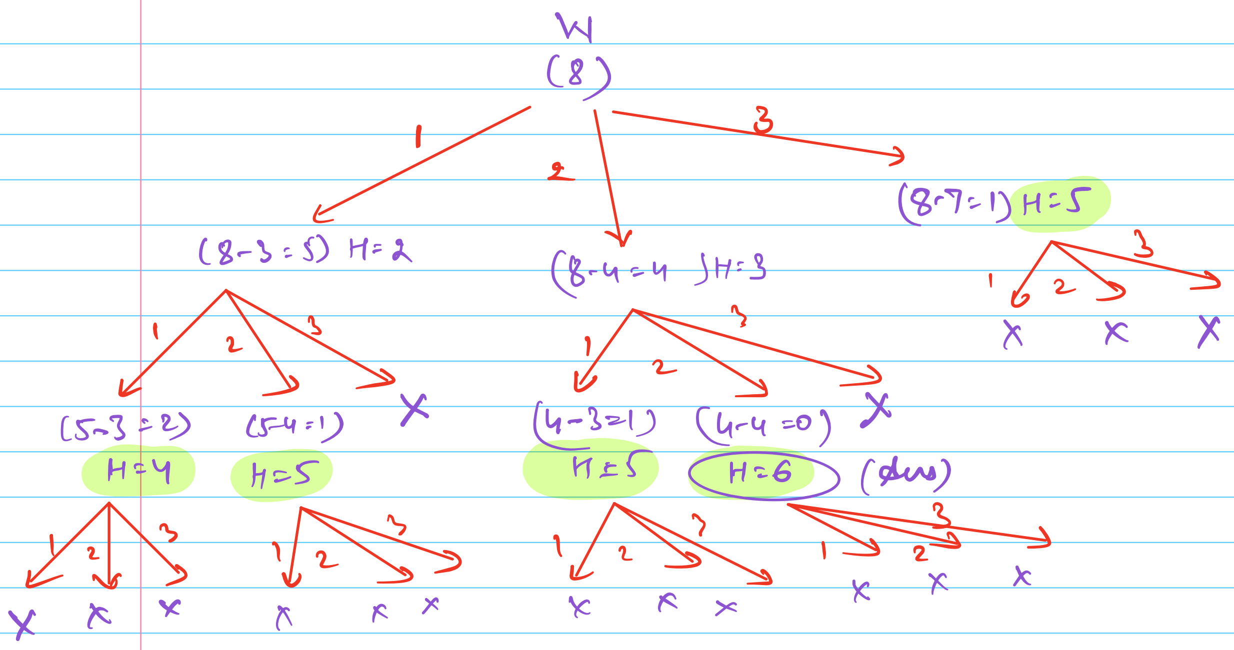
(objects cannot be divided)
(one object can be selected multiple times)

Q: Given N toys with their happiness & weight. Find max total happiness that can be kept in a bag with capacity = W (toys cannot be divided, same toy can be selected multiple times)

$N = 3$
$W = 8$

$$\begin{array}{cccc} & 1 & 2 & 3 \\ h = & [\,2 & 3 & 5\,] \\ w = & [\,3 & 4 & 7\,] \\ & & \times 2 & \end{array}$$

$H = 6$



capacity

dp[i] → max happiness that can be achieved if capacity = i

$dp[0] = 0$

capacity.

$$dp[i] = \max\left(h[j] + dp[i - wt[j]]\right)$$

∀ toys (j)

∀ i, dp[i] = 0

```
for i → 1 to W {
    for j → 1 to N {
        if( i >= wt[j]) {
            dp[i] = max(dp[i], h[j] + dp[i-wt[j]])
        }
    }
}
```

TC: O(N*W)

SC: O(W)

return dp[W];

N = 3

W = 8

j →	1	2	3
h =	[2	3	5]
w =	[3	4	7]

0	1	2	3	4	5	6	7	8	
0	0	0	2	3	3	4	5	6	

Doubts