DP – 3

amit khandelwal
Bhaveshkumar
Burhan
Gagan Kumar S
Gowtham
Hemant Kumar
Nikhil Pandey
Purusharth A
Rajat Sharma
Rajendra
Sanket Giri
Saurabh Ruikar
Shani Jaiswal
shilpa mamillapalli
Sridhar Hissaria
Sumit Adwani
Vimal Kumar

content

- Fractional knapsack
- 0/1 knapsack
- 0/∞ knapsack
- Flip Array { If time permits }

Contest 5 26th

Generalised knapsack

Given N objects with their values Vi profit/loss their weight Wi. A bag is given with capacity W that can be used to carry some objects such that the total sum of object weights W and sum of profit in the bag is maximized or sum of loss in the bag is minimized.

We will try Knapsack when these combinations are given:

- number of objects will be N
- every object will have 2 attributes namingly value and weight
- and capacity will be given C

Fractional Knapsack

Given N cakes with their happiness & weight.
Find max total happiness that can be kept in a bag with capacity C {cakes can be divided}

N = 5

C = 40

	0	1	2	3	4
H →	3	8	10	2	5
W →	10	4	20	8	15

	0	1	2	3	4
H →	3	8	10	2	5
W →	10	4	20	8	15

ans = 23.3

W	H
20	10
4	8
15	5
1	0.3
40	23.3

H	10	20	30	40
W	1	1	1	1

C = 2

~~Always take cakes with max Happiness~~

H	15	20	30
W	1	1	2

C = 2

Idea 2 $\longrightarrow$ Take max happiness / unit weight

		0	1	2	3	4
H	$\longrightarrow$	3	8	10	2	5
W	$\longrightarrow$	10	4	20	8	15

H/W			
0	0.3 [4]	$\longrightarrow$	1 kg {fractional cake}
1	2 [1]	$\longrightarrow$	4
2	0.5 [2]	$\longrightarrow$	20
3	0.25		
4	0.33 [3]	$\longrightarrow$	15
			40

To sort the given input based on happiness per weight

keep taking the cakes while capacity > 0

TC : $O(N \log N)$

SC : $O(1)$

Hypothetical Situation

Bundles of 1000 Rs

{ ratings are high , cost of product }

0/1 Knapsack *** {object cannot be divided}

Given N toys with their happiness & weight.
Find max total happiness that can be kept in a bag with capacity = C {toys cannot be divided}

			0	1	2	3	
N = 4	H	⟶	4	1	5	7	ans = 9
C = 7	W	⟶	3	2	4	5	

Sort based on H/W

0	1.33	H = 7 + 1 = 8
1	0.5	2−2 = 0
2	1.25	
3	1.4	7−5 = 2

Brutefarce Generate all possible subset

Check if the generate subset is within the given capacity

Store the max happiness for all such subsets.

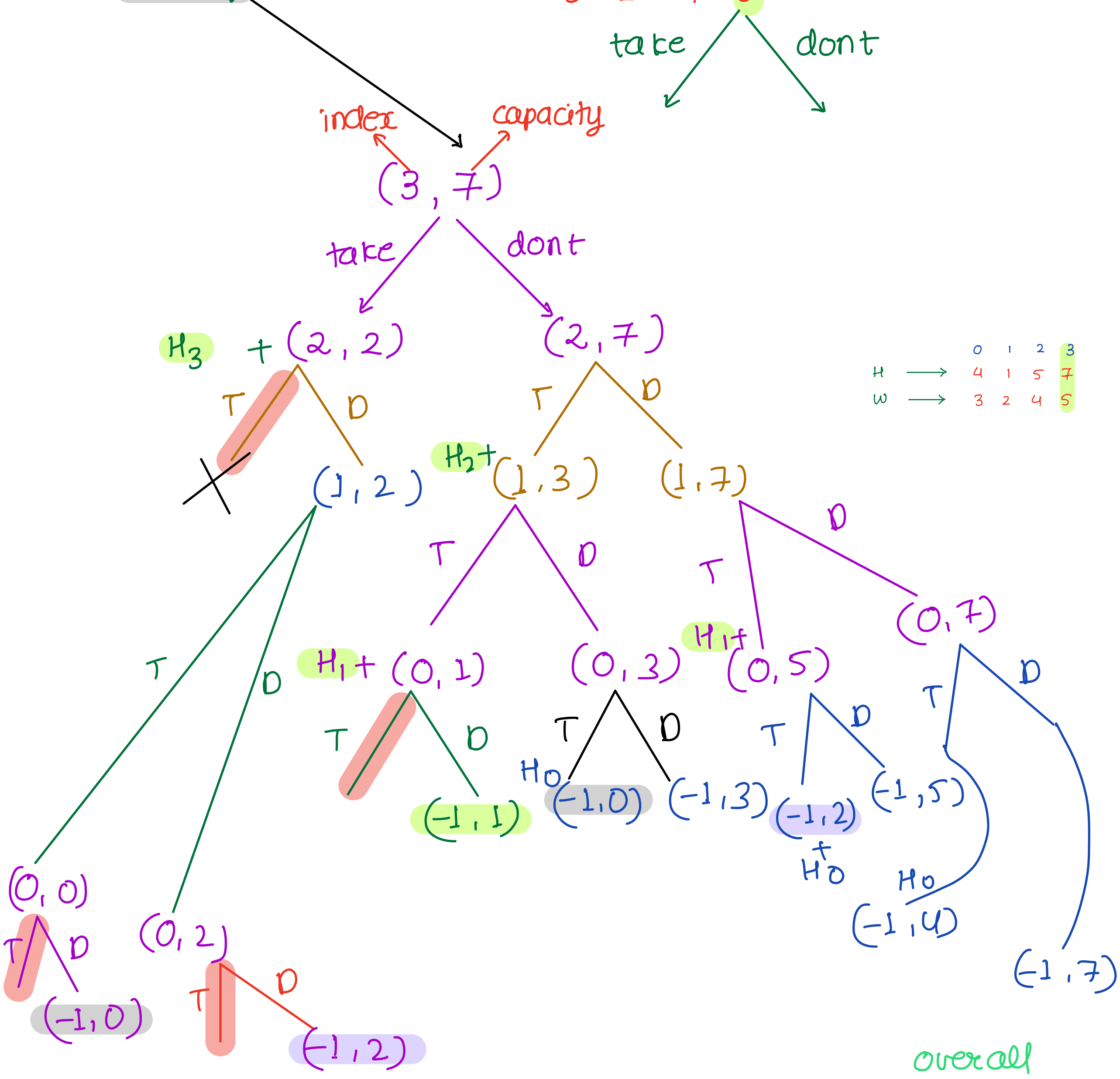
Total subsets = 2^N

TC : $O(N \cdot 2^N)$

N = 4

C = 7

	0	1	2	3
H →	4	1	5	7
W →	3	2	4	5



0 to N-1   0 to C

DP state → (index , capacity)

DP [N] [C+1] → OR Hashmap

overall unique states = N * (C+1)

dp [index] [capacity] — max

- Take → H[index] + index −1 , capacity − W[index]
- dont → index−1 , capacity

Pseudocode

```
int  KS01 ( H[] , W[] , C ) {          C → capacity
       dp = new int[N][C+1]            dp → global
       return  solve ( N−1 , C )
}
```

TC : O(NC)

SC : O(NC)

```
int  solve ( index , capacity ) {
       // Base condition
       if( index < 0 )  return 0
       if (dp[index][capacity] != -1)  return dp[index][capacity]
       take = 0
       dont = solve(index-1 , capacity)
       if (capacity >= W[index])
       {
           take = H[index] + solve (index -1 , capacity - W[index]
       }
       dp[index][capacity] = max (take , dont )
       return max (take , dont )
}
```

Break 22 : 37

0/∞ Knapsack {object cannot be divided}

Unbounded KS {same object can be selected multiple times}.

Given N toys with their happiness & weight.
Find max total happiness that can be kept in a bag with capacity = C

{toys cannot be divided}
{infinite toys are available}

C = 10

	0	1	2
H →	2	2	15
W →	3	1	3

ans = 15*3 + 2*1
= 47

capacity = 10

C = 7

	0	1	2
H →	2	3	5
W →	3	4	7

T D

$\underset{\text{index}}{\uparrow}$ $\underset{\text{capacity}}{\uparrow}$

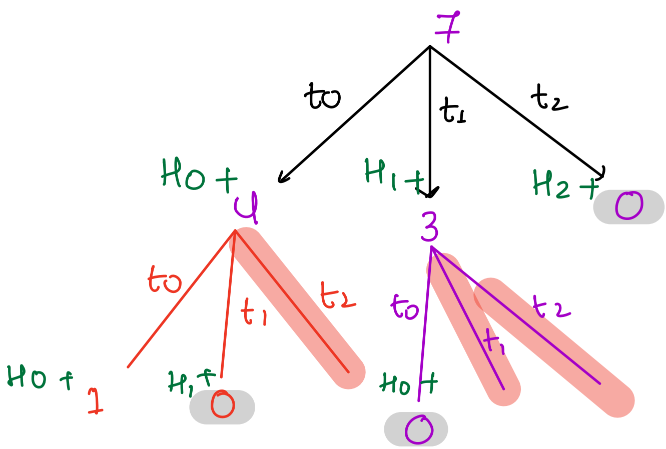
dp[i][C] $\xrightarrow{\text{take } i^{th} \text{ toy}}$ dp[i][C − W[i]]

dp[i][C] $\xrightarrow{\text{dont take } i^{th} \text{ toy}}$ dp[i−1][C]

TC : O(NC) SC : O(NC)

Another way

		0	1	2
	$H \longrightarrow$	2	3	5
$C = 7$	$W \longrightarrow$	3	4	7



dp state $\longrightarrow$ capacity

DP table $\longrightarrow$ dp[C+1]

$$dp[capacity] = \max \begin{cases} \xrightarrow{t_0} H_0 + dp[capacity - W_0] \\ \xrightarrow{t_1} H_1 + dp[capacity - W_1] \\ \xrightarrow{t_2} \\ \vdots \\ \xrightarrow{t_{n-1}} H_{n-1} + dp[capacity - W_{n-1}] \end{cases}$$

Pseudocode

dp[C+1] // -1

```
int solve ( capacity ) {
    if( dp[capacity] != -1)  return  dp[capacity]
    MH = 0
    for i ⟶ 0 to N-1 {
        if ( capacity >= wi ) {
            MH = max (MH , Hi + solve (capacity
                                           - wi )
        }
    }
    dp[capacity] = MH
    return MH
}
```

TC : O (No. of unique dp states * TC per state)
(C+1) N

TC : O (NC)

SC : O (C)

C = 100

V	1 ,	30
W	1 ,	50

$1 \xrightarrow{\text{100 times}} 100$

$50 \xrightarrow{\text{2 times}} 60$

V	1	3
W	1	2

C = 3

Flip Array

Given an array with +ve elements, $\forall i \quad A[i] > 0$
Flip sign { * -1 } of some of its elements such that sum of elements of final array is min non-negative integer.

Find min # elements to flip.

A { 10 15 6 3 3 } ans = 2

$10 - 15 + 6 - 3 + 3 = 1$

A { 2 1 1 } ans = 1

$-2 + 1 + 1 = 0$

sum of selected elements <= sum of not selected elements

+ sum of selected elements   sum of selected elements

2 * sum of selected <= total sum

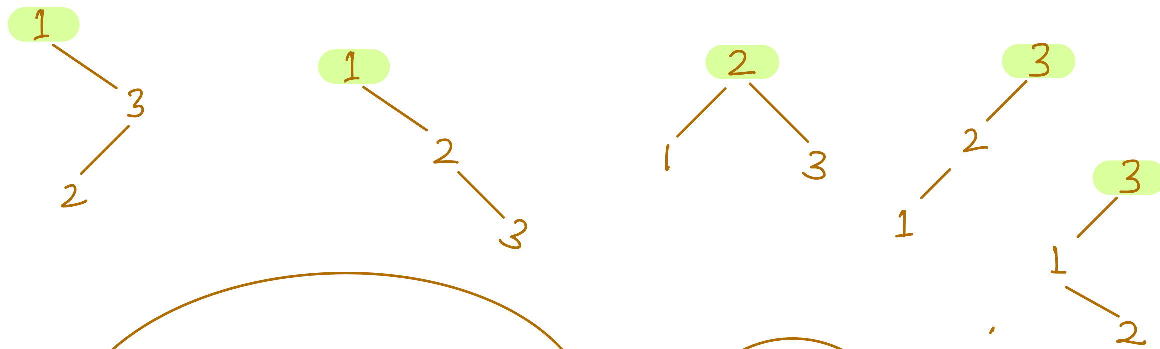
sum of selected <= total sum / 2

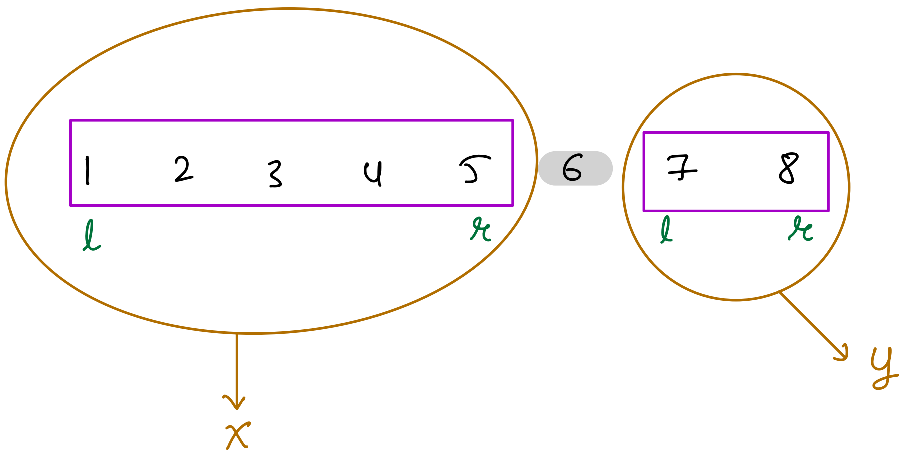
$\underbrace{\text{sum of selected}}_{\text{capacity}}$

selecting i^{th} item gives you 1 happiness

1 2 3

No. of unique BST that can be formed





Total no. of BST rooted at 6 = $x * y$

$f(l, r) \longrightarrow$ No. of unique BST arrangements from L to R

$f(l\ r)$

root at index l

root at index $l+1$

⋮

root at index r

Sum

catalan NO.

$$\frac{1}{n+1} \; {}^{2n}C_n$$

$n = 3$

$$\frac{1}{4} * {}^{6}C_{3} = \frac{\cancel{6} * 5 * \cancel{4}}{\cancel{4} * \cancel{3!}}$$

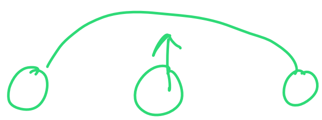
$$= \boxed{5}$$

---

0 – ∞ knapsack always select toy based on H/W

4 6

2 $C = 10$



1 5 10

10 5 10 104