Knapsack Problem

02 April 2025 00:34

sometimes D.P > greedy , sometimes greedy > D.P (> : preffered)

0-1 KNAPSACK PROBLEM

PROBLEM :→ n items , V_i → money , W_i → weight [thief robbing a store]

W → max. weight in Knapsack.

Find max. profit.

→ let profit per item be p_i

Motive :→ $\sum_{i \in T} p_i$ is maximized (given $\sum_{i \in T} W_i \leq W$)

W = 50

i	1	2	3	4	5	6
Wi	10	15	20	25	05	10
Pi	100	200	100	210	50	210

* Brute force → catch every element and try to fit → $O(n^2)$
* Combination of all values → $O(2^n)$ → 0/1 So for n ——— → 2^n possible

* Adding another weight w, weight of the subproblem.

then to find P[K, W]

$$P[K,W] = \begin{cases} P[K-1, W] & \text{if } W_K > W \\ \max(P[K-1, W], P[K-1, W-W_K] + P_K) & \text{if } W_K \leq W \end{cases}$$

W_K → weight of object; W → KNAPSACK weight. → item cannot be included.

$P[K-1, W-W_K]$ → profit without element K

P_K → ⊕ profit of element K.

TABULATION METHOD.

no item available → W, W_K

Weight	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...	20	21	22	23	...	30	31	32	...	40	49	...	50
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0		0	0	0		0	0		0
10	0	0	0	0	0	0	0	0	0	60	60	60			60			60				60				60			60
20	0	0	0	0	0	0	0	0	0	60					60			100				100				100			100
30	0	0	0	0						60	60				60			100				100				160			160
																										180 (40)			220

C=50	w1	p1	w2	p2	w3	p3
N=3	10	60	20	100	30	120

Weights (kg)	Knapsack capacities (kg)										
2 1 5 3	0	1	2	3	4	5	6	7	8	9	10
	0	0	0	0	0	0	0	0	0	0	0
	0	0	300	300	300	300	300	300	300	300	300
	0	200	300	500	500	500	500	500	500	500	500
	0	200	300	500	500	500	600	700	900	900	900
	0	200	300	500	700	800	1000	1000	1000	1100	1200
300 200 400 500											
Values ($)											

Maximum Value in Knapsack: $ 1200

kg :→ $m = 8$ $p = \{1, 2, 5, 6\}$

$n = 4$ $w = \{2, 3, 4, 5\}$

		V	0	1	2	3	4	5	6	7	8
P	W	0	0	0	0	0	0	0	0	0	0
1	2	1	0	0	1	1	1	1	1	1	1
2	3	2	0	0	1	2	2	3	3	3	3
5	4	3	0	0	1	2	5	5	6	7	7
6	5	4	0	0	1	2	5	6	6	7	(8)

$V[i, w] = \max\left(V[i-1, w],\ V[i-1, w-w[i]] + P[i]\right)$

// $w - w[i] < 0$ if $w[i] > w$

Hence, $V[i-1, w]$ accepted.

Hence the poem $V[i, w] = \begin{cases} V[i-1, w] & ; w[i] > w \\ \max\left(V[i-1, w],\ V[i-1, w-w[i]] + P_K\right) & ; \text{otherwise} \end{cases}$

we either take the object or combination of objects.

Solution :→

x_1	x_2	x_3	x_4
0	1	0	1

→ check the upper element if not present, change is due to the element

Reason, $8 \rightarrow$ 6 corresponds to 5 so 5 (i.e. 4th)

$8 - 6 = 2$ corresponds to 3rd (X)

2 corresponds to 2nd (✓)

$8 - 6 - 2 = 0$ corresponds to 0th not 1st.

∴ 2nd & 4th object.

```
for (int i = 0; i <= n; i++)
    for (int w = 0; w <= C; w++)   // C is the capacity.
    {
        if (i == 0 || w == 0)
            m[i][w] = 0;
        else if (w[i] > w[j])
            m[i][w] =   [i-1][w];
        else
            m[i][w] = max(m[i-1, w[j]],  [i-1, w[j] - w[i] + P[i]);
```

```
else
    m[i][w] = max(m[i-1, w[j]] ,  [i-1, w[j] - w[i] + p[i]) ;
}
cout << m[n][c];
```