Dynamic Programming

0/1 Knapsack problem

Knapsack - W units (capacity)

$v \in \mathbb{R}^n$ $w \in \mathbb{R}^n$

$s \in \{0,1\}^n$

$o_1, o_2, \ldots o_n$

$w_1 \; w_2 \quad w_n$

$v_1 \; v_n \quad v_n$

$s : [1, 0, 1, 0, 0]$

$o_1 \quad o_3$

maximize $\vec{s} \cdot \vec{v}$

sub to: $\vec{s} \cdot \vec{w} \leq W$

max. $\sum_{i=1}^{n} s_i v_i$ sub to $\sum_{i=1}^{n} s_i w_i \leq W$

	o_1 ✓	o_2 ×	o_3 ×
w	6	5	5
v	8	5	5

10 kg

4 kg

6 kg

o_1

10 kg

8 units

(i) Structure of the memorization table.

(ii) figure out an expression to fill the table.

(iii) $T[m,n] \rightarrow$ last row, last col of the table contains the solution

(v) for finding $\vec{s}$, we backtrack the table and retrace the steps.

T — Capacity of k/s

	0	1	2	3	4	5	6	7	8	9	10
o_1 $w_1=1$, $v_1=1$	0	1	1	1	1	1	1	1	1	1	1
o_2 $w_2=2$ $v_2=6$	$\boxed{0}$	1	$\boxed{6}$	7	7	7	7	7	7	7	7
o_3 $w_3=5$, $v_3=18$	0	1	6	7	7	18	19	24	25	25	25
o_4 $w_4=6$, $v_4=22$	0	1	6	7	7	18	22	24	28	29	29
o_5 $w_5=7$, $v_5=28$	0	1	6	7	7	18	22	28	29	34	35

"consider all out-of-bounds entries as $-\infty$"

Value / profit obtained from k/s

$T[i,j] \leftarrow$ max profit that can be obtained if k/s has a capacity of j kg and if only objects $1, 2, \ldots i$ are available for filling the k/s

$T[i,j] = \max(T[i-1, j], T[i-1, j-w_i] + v_i)$

excluding object i — including object i

$T[2,2] \leftarrow \max(T[1,2], T[1, 2-2] + 6)$

$\leftarrow \max(1, 6)$

$\leftarrow 6$

$T(n, W) = O(nW)$ - for filling the table.

$T(n) = O(n)$ - for backtracking

$S = [1, 1, \cancel{0}, \cancel{0}, 1]$

Back Tracking

(1) $T[5,10] \leftarrow \max(T[4,10], T[4, 10-7] + 28)$

$\leftarrow \max(29, 35)$

(2) $T[4,3] \leftarrow \max(T[3,3], T[3, 3-6] + 22)$

$\leftarrow \max(7, -\infty)$

(3) $T[3,3] \leftarrow \max(T[2,3], T[2, 3-5] + 18)$

(4) $T[2,3] \leftarrow \max(T[1,3], T[1, 3-2] + 6)$

$\leftarrow \max(1, 7)$. 7

(5) $T[1,1] \leftarrow \max(T[0,1], T[0, 1-1] + 6)$

(6) $T[0, \boxed{0}]$ ×

M_1 - 13×5

M_2 - 5×89

M_3 - 89×3

M_4 - 3×34

$13 \times 5 \times 89 + 89 \times 3 \times 34 + \quad 13 \times 89 \times 34$

$\underset{13 \times 34}{P} = \underbrace{(M_1 . M_2)}_{I_1 \; (13 \times 89)} \cdot \underbrace{(M_3 . M_4)}_{I_2 \; (89 \times 34)} : O_1$

$M_1 ((\underbrace{M_2 . M_3}_{I}) M_4) : O_2$

O_1

#scalar xs } :

$()_{a \times b} \times ()_{b \times c}$

O_2

#scalar xs } $= 13 \times 5 \times 34 + 5 \times 89 \times 3 + 5 \times 3 \times 34$

$M_1 ((M_2 M_3) M_4)$ — 13×5, 5×34

1. $((M_1 M_2) M_3) M_4$ —
2. $(M_1 M_2)(M_3 M_4)$ —
3. $(M_1 (M_2 M_3)) M_4$ — least # of scalar multiplications
4. $M_1 ((M_2 M_3) M_4)$ —
5. $M_1 (M_2 (M_3 M_4))$ —

$[(M_1 (M_2 (\ldots M_i)] [(M_{i+1} \ldots M_n)]$

$T(n) = \sum_{i=1}^{(n-1)} T(i) \, T(n-i) \Leftarrow$ Catalan Recursion

(Dynamic programming helps)!

n	1	2	3	4	5	 10	. . . 15
T(n)	1	1	2	5	14	4862	2674440

Brute force!

↑ Catalan numbers.