

Change-making problem

3.31.2017

d_1, d_2, \dots, d_n

N - amount to be changed

$c[i, j]$ - min. no. of coins needed to pay an amount j using coins with denominations $\leq i$

we can use only coins d_1, d_2, \dots, d_i

the amount to be changed is j

- objective is to compute $c[n, N]$

$$c[i, j] = \min(1 + c[i, j - d_i], c[i-1, j])$$

example

$N = 8$

coin denominations

d

1	2	3
1	4	6

$n = 3$

$c[3, 8]$ - min no. of coins used to change $N = 8$

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

$c[i, j]$: if $i = 1$ and $j < d_1 \Rightarrow \infty$

else if $i = 1$ then $1 + c[1, j - d_1]$

elseif $j < d_i$ then $c[i-1, j]$

else $\min(c[i-1, j], 1 + c[i, j - d_i])$

$$\underline{i=1} \quad d_1''^1$$

$$c[1,1] = 1 + c[1,0] = 1$$

$$c[1,2] = 1 + c[1,1] = 2$$

$$c[1,3] = 1 + c[1,2] = 3$$

...

$$\underline{i=2} \quad d_1''^1, d_2''^4$$

$$c[2,1] = c[1,1] = 1$$

$$c[2,2] = c[1,2] = 2$$

$$c[2,3] = c[1,3] = 3$$

$$c[2,4] = \min(c[1,4], \underline{1 + c[2,0]}) = 1$$

$$c[2,5] = \min(c[1,5], \underline{1 + c[2,1]}) = 2$$

$$c[2,6] = \min(c[1,6], \underline{1 + c[2,2]}) = 3$$

$$c[2,7] = \min(c[1,7], \underline{1 + c[2,3]}) = 4$$

$$c[2,8] = \min(c[1,8], \underline{1 + c[2,4]}) = 2$$

$$\underline{i=3} \quad d_1''^1, d_2''^4, d_3''^6$$

$$c[3,1] = c[2,1] = 1$$

$$c[3,6] = \min(c[2,6], \underline{1 + c[3,0]}) = 1$$

$$c[3,7] = \min(c[2,7], \underline{1 + c[3,1]}) = 2$$

$$c[3,8] = \min(\underline{c[2,8]}, 1 + c[3,2]) = 2$$

The minimum number of coins needed to change the amount $N=8$ is $c[3,8]=2$

Print Coins (c, 3, 8)

Print Coins (c, 2, 8)

Print Coins (c, 2, 4) print d_2

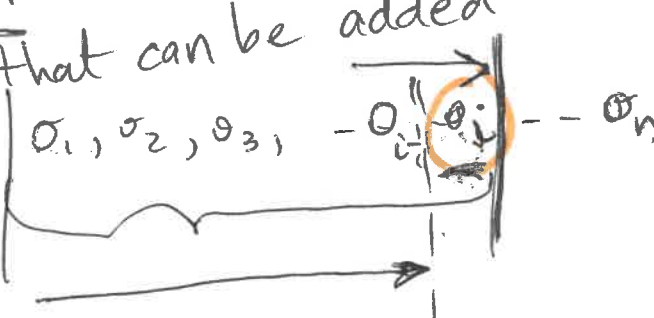
Print Coins (c, 2, 0) print d_2

Optimally, we change $N=8$ using the coins $\{d_2, d_2\}$

Dynamic Programming

0-1 Knapsack Problem

$V[i, j]$ = max value that can be added
select from objects o_1, o_2, \dots, o_i
capacity constraint of the knapsack



$V[1..n, 0..W]$

- objective: compute $V[n, W]$

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

if $v_i + V[i-1, j-w_i] \geq V[i-1, j]$ then the object o_i has been added to the knapsack.

example

$n=5$ objects

$W=11$

object	Weight	Value
1	1	1
2	2	6
3	5	18
4	6	22
5	7	28

$V[1..5, 0..11]$

Weight limit is j

object i

	0	1	2	3	4	5	6	7	8	9	10	11
1	0	1	1	1	1	1	1	1	1	1	1	1
2	0	1	6	7	7	7	7	7	7	7	7	7
3	0	1	6	7	7	18	19	24	25	25	25	25
4	0	1	6	7	7	18	22	24	28	29	29	40
5	0	1	6	7	7	18	22	28	29	34	35	40

$$\begin{cases} V[i, 0] = 0 \text{ for all } i \\ V[i, j] = \max(V[i-1, j], v_i + V[i-1, j-w_i]) \\ V[0, j] = 0 \\ V[i, j] = -\infty \text{ if } j < 0 \end{cases}$$

$i=1$ σ_1

$$V[1, 1] = \max(V[0, 1], 1 + V[0, 0]) = 1$$

$$V[1, 2] = \max(V[0, 2], 1 + V[0, 1]) = 1$$

...

$i=2$ σ_1, σ_2

$$V[2, 1] = \max(V[1, 1], 6 + V[1, -1]) = 1$$

$$V[2, 2] = \max(V[1, 2], 6 + V[1, 0]) = 6$$

$$V[2, 3] = \max(V[1, 3], 6 + V[1, 1]) = 7$$

...

$i=3$ $\sigma_1, \sigma_2, \sigma_3$

$$V[3, 5] = \max(V[2, 5], 18 + V[2, 0]) = 18$$

...

$i=4$ $\sigma_1, \sigma_2, \sigma_3, \sigma_4$

$$V[4, 6] = \max(V[3, 6], 22 + V[3, 0]) = 22$$

$$V[4, 11] = \max(V[3, 11], 22 + V[3, 5]) = 40$$

$i=5$ $\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5$

$$V[5, 11] = \max(V[4, 11], 28 + V[4, 4]) = 40$$

$V[n, W] = V[5, 11] = 40 \Rightarrow$ max value of the objects in the knapsack for an optimal solution

Print Objects ($V, 5, 11$)

Print Objects ($V, 4, 11$)

Print Objects ($V, 3, 5$)

Print Objects ($V, 2, 0$)

print σ_4

print σ_3

Optimal solution contains objects $\{\sigma_3, \sigma_4\}$

total value = 40

total weight = 11 $\leq W = 11$

Sequence Alignment problem

X: ^{1 2 3 4}
a b d c
Y: ^{1 2 3 4 5}
a c a a d

(1,1) (3,2) (4,3) is NOT a matching

X: ^{1 2 3 4 5 6 7}
a b a c d e f
Y: ^{1 2 3 4 5}
b a c c e

(1,2) (2,1) (3,3) (4,4) (5,5) (6,6) (7,7) is a matching

Alignment: no crossing pairs are allowed!

X: ^{1 2 3 4 5 6 7}
a b a c d e f
Y: ^{1 2 3 4 5}
b a c c e

alignment: (1,2) (2,1) (3,3) (4,4) (5,5) (6,6) (7,7)

*for any pairs (i,j) and (i',j') ,
if $i < i'$ then $j < j'$

X = $x_1 x_2 \dots x_i$
Y = $y_1 y_2 \dots y_j$

$x_1 \dots x_m$
 $y_1 \dots y_n$