Algorithmn HW9

5140379032 JIN YI FAN

Problem 7.5

i\j	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1	1
2	0	1	1	1	1	2	2	2
3	0	1	1	2	2	2	2	2
4	0	1	1	2	2	3	3	3
5	0	1	1	2	2	3	3	4
6	0	1	1	2	3	3	3	4
7	0	1	2	2	3	3	4	4

The figure on the left shows the result matrix of the DP.
So the length of longest common subsequence is 4, and the longest common subsequence of $xzyzzyx$ and $zxyyzxz$ is $zyzz$.

Problem 7.11

C[1,1]=0	C[1,2]=36	C[1,3]=84	C[1,4]=96	C[1,5]=124
	C[2,2]=0	C[2,3]=72	C[2,4]=84	C[2,5]=126
		C[3,3]=0	C[3,4]=48	C[3,5]=132
			C[4,4]=0	C[4,5]=56
				C[5,5]=0

(a). According to the figure above, we can conclude that the minimum number of scalar multiplications needed is $C[1,5] = 124$.
(b). To achieve optimal situation, the order should be $M_1 \times (M_2 \times (M_3 \times M_4)) \times M_5$

Problem 7.26

The running time will be $\Theta(n\lfloor C/K \rfloor)$
Counterexample: Suppose $C = 12$

s_i	1	11	10
v_i	1	1	1

when $K = 6$, it will be $C = 2$ and

$\lfloor s_i/K \rfloor$	0	1	1
v_i	1	1	1

Obviously, some s_i become 0 and meaningless.

Problem 7.30

(a)

Require: total value y and a sequence of coin value $\{v_1 = 1, v_2 \dots v_n\}$
Ensure: the pay plan $\{x_1, x_2 \dots x_n\}$

$dp[j \leftarrow 0 \text{ to } y] \leftarrow (j == 0?\ 0 : -1)$
for $i \leftarrow 1$ to n **do**
    for $j \leftarrow v_i$ to y **do**
        $dp[j] = min\{dp[j], dp[j - v_i] + 1\}$ ▷ find the minimum volume
        $take[j] = i$ if $dp[j] == dp[j - v_i] + 1$ ▷ store the adding coin
    end for
end for
$j \leftarrow y$
while $dp[j]! = -1$ **do** ▷ recover the plan
    $x_{take[j]} \leftarrow x_{take[j]} + 1$
    $j \leftarrow j - v_{take[j]}$
end while

(b) Time complexity is $O(y \cdot n)$, space complexity is $O(y)$

(c) Problem 7.27, the complete knapsack problem is to **maximize** *value* with *volume* constraint, while this problem is to **minimize** *volume*(which is 1 for each) with *value* constraint.

Problem 7.34

Require: a DAG $G = (V, E)$, start point s and end point t
Ensure: a longest path $P = \{s \dots t\}$

TOPOSORT(V)
$dp[s] \leftarrow 0$ ▷ $dp[i]$ means the longest path end with i
for each $v \in V$ s.t. $s < v \leq t$ **do**
  $dp[v] \leftarrow max_{(u,v)\in E}\{dp[u] + w(u, v)\}$ ▷ check every edge connected
  $pred[v] \leftarrow u$ s.t. $w(u, v)$ is maximum ▷ store the predecessor of each vertex
end for
return $dp[t]$

SubsetSum

Require: a list of n positive integers $a_1, a_2 \dots a_n$, a positive integer t.
Ensure: whether there is some subset of a_is add up to t

for $i \leftarrow 0$ to n **do**
  $T[i, 0] \leftarrow 0$ ▷ $T[i, s]$ is the boolean value of 'does a subset $\{a_1 \dots a_i\}$ add up to s?'
end for
for $i \leftarrow 1$ to n **do**
  for $s \leftarrow 1$ to t **do**
    $T[i, s] \leftarrow T[i - 1, s] \mid T[i - 1, s - a_i]$
  end for
end for
return $T(n, t)$

PickCards

(a) Consider the sequence $2, 100, 1, 1$, using *greedy* strategy, the first player will take the front 2, and the second player will take 100 and win the game.

(b)

$dp[i][0] \leftarrow 0$ for each $i \in range(0, n + 1)$
$dp[0][j] \leftarrow 0$ for each $j \in range(0, n + 1)$
$dp[i][i] \leftarrow s_i$ for each $i \in range(1, n + 1)$
for $i \leftarrow 1$ to n **do**
  for $j \leftarrow i + 1$ to n **do**
    $first \leftarrow min(dp[i + 2][j], dp[i + 1][j - 1]) + s_i$ ▷ p1: take first, p2:min(take first, take last)
    $last \leftarrow min(dp[i][j - 2], dp[i + 1][j - 1]) + s_j$ ▷ p1: take last, p2:min(take last, take first)
    $pred[i][j] \leftarrow first > last?\ s_i : s_j$
    $dp[i][j] \leftarrow max(first, last)$
  end for
end for ▷ Then player can refer to the *pred* matrix to make the optimal decision for each step