UWI

Design and Analysis of Algorithms

Tutorial 7- Sample Solutions

Dynamic Programming

It is important that we learn the theory behind dynamic programming.

1. Explain the theory behind the transform and conquer approach

   (a) What is meant by the transform and conquer approach?

   DP is a technique for solving problems with overlapping sub-problems. These sub-problems arise from a recurrence relating a given problem's solution to solutions of its smaller sub-problems.

   Rather than solving overlapping sub-problems again and again, dynamic programming suggests solving each of the smaller sub-problems only once and recording the results in an array (1D or 2D array) from which a solution to the original problem can then be obtained.

   (b) What are the two approaches of dynamic programming?

   1. The classic bottom-up DP approach solves all smaller sub-problems of a given problem.
   2. The top-down DP approach avoids solving unnecessary sub-problems.

   (c) What is the principle of optimality?

   The principle of optimality says that an optimal solution to any instance of an optimization problem is composed of optimal solutions to its sub-instances.

2. Solve the coin row problem using dynamic programming for the coin row {10, 5, 1, 2, 5, 1}

   The coin row problem's goal is to pick up the maximum amount of money subject to the constraint that no two coins adjacent in the initial row can be picked up. Let $F(n)$ be the maximum amount that can be picked up from n coins. We have the following recurrence:

$$F(n) = \max\{c_n + F(n–2), F(n–1)\}, \text{ for } n > 1 \tag{1}$$

$$F(1) = c_1 \tag{2}$$

$$F(0) = 0 \tag{3}$$

   We proceed by calculating the value of F where $n = \{0, 1, \ldots, 5, 6\}$

$$F(0) = 0 \tag{4}$$

$$F(1) = c_1 = \mathbf{10} \tag{5}$$

$$F(2) = \max\{c_2 + F(2–2), F(2–1)\} = \max\{5 + 0, 10\} = 10 \tag{6}$$

$$F(3) = \max\{c_3 + F(3–2), F(3–1)\} = \max\{\mathbf{1 + 10}, 10\} = 11 \tag{7}$$

$$F(4) = \max\{c_4 + F(4–2), F(4–1)\} = \max\{2 + 10, 11\} = 12 \tag{8}$$

$$F(5) = \max\{c_5 + F(5–2), F(5–1)\} = \max\{\mathbf{5 + 11}, 12\} = 16 \tag{9}$$

$$F(6) = \max\{c_6 + F(6–2), F(6–1)\} = \max\{1 + 12, 16\} = 16 \tag{10}$$

i	0	1	2	3	4	5	6
c_n	-	10	5	1	2	5	1
$F(n)$	0	10	10	11	12	16	16

Table 1: Dynamic programming calculations of the coin row problem

To figure out the solution we must backtrack from $F(6)$ to determine the optimal solution. We notice that $F(6)$ just takes $F(5)$'s solution, so we backtrack to there. $F(5)$ sums the coin at index 5 with $F(3)$'s solution. So we go to $F(3)$, where we see that it sums the coin at index 3 with the solution from $F(1)$. $F(1)$'s solution is simply the coin at index 1. Thus, we can determine that the optimal solution contains the coins at indexes 1, 3 and 5, which have the values 10, 1 and 5 respectively.

3. Solve the change making problem using dynamic programming for the coin denominations {1, 2, 5} and where $n = 8$.

   The change making problem's goal is to give change for an amount n using the minimum number of coins of denominations. We assume that the quantities of coins for each denomination is unlimited. Let $F(n)$ be the minimum number of coins whose values add up to n, we have the following recurrence:

$$F(n) = \min\{F(n–d_j)\} + 1, \text{ for } n > 0 \text{ and } d_j \leq n \tag{11}$$

$$F(0) = 0 \tag{12}$$

   We proceed by calculating the value of F where $n = \{0, 1, \ldots, 7, 8\}$

$$F(0) = 0 \tag{13}$$

$$F(1) = \min\{F(1–1)\} + 1 = \min\{0\} + 1 = 1 \tag{14}$$

$$F(2) = \min\{F(2–1), F(2–2)\} + 1 = \min\{1, 0\} + 1 = 1 \tag{15}$$

$$F(3) = \min\{F(3–1), F(3–2)\} + 1 = \min\{1, 1\} + 1 = 2 \tag{16}$$

$$F(4) = \min\{F(4–1), F(4–2)\} + 1 = \min\{2, 1\} + 1 = 2 \tag{17}$$

$$F(5) = \min\{F(5–1), F(5–2), F(5-5)\} + 1 = \min\{2, 2, 0\} + 1 = 1 \tag{18}$$

$$F(6) = \min\{F(6–1), F(6–2), F(6-5)\} + 1 = \min\{1, 2, 1\} + 1 = 2 \tag{19}$$

$$F(7) = \min\{F(7–1), F(7–2), F(7-5)\} + 1 = \min\{2, 1, 1\} + 1 = 2 \tag{20}$$

$$F(8) = \min\{F(8–1), F(8–2), F(8-5)\} + 1 = \min\{2, 2, 2\} + 1 = 3 \tag{21}$$

n	0	1	2	3	4	5	6	7	8
$F(n)$	0	1	1	2	2	1	2	2	3

Table 2: Dynamic programming calculations of the change making problem

4. Solve the 0/1 knapsack problem using dynamic programming for the instance shown in table 3 with a max knapsack capacity of **3**.

Item	Weight	Value
1	2	\$20
2	1	\$15
3	2	\$25

Table 3: A small instance of the 0/1 knapsack problem

The knapsack problem's goal is given n items of known weights and values and a knapsack of capacity W, find the most valuable subset of the items that fit into the knapsack. We have the following recurrence relation:

$$F(i, j) = \begin{cases} F(i–1, j) & \text{if } w_i > j, \\ \max\{v_i + F(i–1, j–w_i), F(i–1, j)\} & \text{if } w_i \leq j \end{cases} \tag{22}$$

$$F(0, j) = 0 \text{ for } j \geq 0 \tag{23}$$

$$F(i, 0) = 0 \text{ for } i \geq 0 \tag{24}$$

We proceed by calculating the value of F where $i = \{0, 1, 2, 3\}$ and $j = \{0, 1, 2, 3\}$

$$F(1,1) = F(1-1,1) = 0 \tag{25}$$

$$F(1,2) = \max\{v_1 + F(1-1,2-w_1), F(1-1,2))\} = \max\{20+0,0\} = 20 \tag{26}$$

$$F(1,3) = \max\{v_1 + F(1-1,3-w_1), F(1-1,3))\} = \max\{20+0,0\} = 20 \tag{27}$$

$$F(2,1) = \max\{v_2 + F(2-1,1-w_2), F(2-1,1))\} = \max\{\mathbf{15+0},0\} = 15 \tag{28}$$

$$F(2,2) = \max\{v_2 + F(2-1,2-w_2), F(2-1,2))\} = \max\{15+0,20\} = 20 \tag{29}$$

$$F(2,3) = \max\{v_2 + F(2-1,3-w_2), F(2-1,3))\} = \max\{15+20,20\} = 35 \tag{30}$$

$$F(3,1) = F(3-1,1) = 15 \tag{31}$$

$$F(3,2) = \max\{v_3 + F(3-1,2-w_3), F(3-1,2))\} = \max\{25+0,20\} = 25 \tag{32}$$

$$F(3,3) = \max\{v_3 + F(3-1,3-w_3), F(3-1,3))\} = \max\{\mathbf{25+15},35\} = 40 \tag{33}$$

$$\tag{34}$$

	capacity j			
i	0	1	2	3
0	0	0	0	0
1	0	0	20	20
2	0	**15**	20	35
3	0	15	25	**40**

Table 4: A small instance of the 0/1 knapsack problem

To figure out the solution we must backtrack from $F(3,3)$ to determine the optimal solution. We notice that $F(3,3)$ sums item 3's value with the solution from $F(2,1)$ so we backtrack from there. $F(2,1)$ sums item 2's value with the solution from $F(1,0)$ and $F(1,0)$ has no real value. Therefore, the optimal subset of items to include in the knapsack are items 2 and 3.