Christopher Matian
CS325 - Spring 2019
04/14/2019

Problem 1 - Rod Cutting

CS 325 - Homework Problems Week 3

15.1-2) Counter-Example to "Greedy Strategy"

length i	Price p_i	p_i / i
1	1	1
2	20	10
3	33	11
4	36	9

Assume rod length = 4

2 cuts

The greedy algorithm would make the first cut @ length 3 valued at 33. That leaves us a rod length 1 of value 1. The total price would be 34.

The DP cut would cut the rod into lengths of 2 each for a total value of 40 dollars.

15.1-3) Rod-cutting (

Problem 2 - Modified Rod Cutting

15.1-3)

```
rod-cutting (p, n, c):
    Some array r[0:n]
    r[0] = 0
    for j = 1 to n:
        Q = p[j]
  ①     for i = 1 to j-1:
  ②         Q = MAX(Q, p[i] + r[j-i] - c)
        r[j] = Q
    return r[n]
```

2 modifications: ① inner loop runs from i=1 to j-1 instead of j

② Adjust MAX to reflect cost (c) of each cut.

Problem 3 - Making Change

a) Pseudo-code for a dynamic programming solution:

```
makechange(n)
   C[<0] = ∞
   C[0] = 0
      for p = 2 to n
         If min = ∞
            for i = 1 to k
               if p >= d`
                  if c[p - d`] + 1
                     coin = i
            C[p] = min
            S[p] = coin
```

b) Run time of the coin-change DP algorithm should be O(nK).

Problem 4 - Shopping Spree

a) Pseudo-code for programming solution:

```
shopping(data)
T = # of test cases
table = 2d array to store items to hold
for each T
      N = # of items
      for each N
            weightArray = w
            priceArray = p
      F = # of family members
      for each F
            knapsackAlgorithm(w, p, n, mw, table)
      output information to outfile
```

b) The bones of the algorithm is essentially just a knapsack algorithm that was provided to us in the lecture notes. Ultimately, it's pseudo-polynomial and would be somewhere in the ballpark of O(nK).