# Homework 5

1. Coin-Row Problem

C = [ 5, 1, 2, 10, 6 ]

F0 = 0, f1 = 5, fi = max (ci + fn-2, fi-1)

i = 2: max (1 + F(0), F(1)) => max (1 , 5) == 5 cents

i = 3: max (2 + F(1), F(2)) => max (7 , 5) == 7 cents

i = 4: max (10 + F(2), F(3)) => max (15 , 7) == 15 cents

i = 5: max (6 + F(3), F(4)) => max (13, 15) == 15 cents

Max change we can get is 15 cents picking the 1st and 4th coins

1. Change-Making Problem (find all solutions)

N=9, C = [ 1, 3, 5 ]

N0 = 0, i 1>n, j 1>m and I > dj, temp = min (n(i-dj), temp)

i = 1: N(i) = temp +1 => 1 coin

j = 1: temp = min(N(i – dj), temp) => min(0 , inf) => 0

j = 2: temp = min(N(i – dj), temp) => min(na , 0) => 0

j = 3: temp = min(N(i – dj), temp) => min(na , 0) => 0

i = 2: N(i) = temp +1 => 2 coins

j = 1: temp = min(N(i – dj), temp) => min(N(1) , inf) => 1

j = 2: temp = min(N(i – dj), temp) => min(na , 1) => 1

j = 3: temp = min(N(i – dj), temp) => min(na , 1) => 1

i = 3: N(i) = temp +1 => 1 coin

j = 1: temp = min(N(i – dj), temp) => min(N(2) , inf) => 2

j = 2: temp = min(N(i – dj), temp) => min(N(0) , 2) => 0

j = 3: temp = min(N(i – dj), temp) => min(na , 0) => 0

i = 4: N(i) = temp +1 => 2 coins

j = 1: temp = min(N(i – dj), temp) => min(N(3) , inf) => 1

j = 2: temp = min(N(i – dj), temp) => min(N(1) , 1) => 1

j = 3: temp = min(N(i – dj), temp) => min(na , 1) => 1

i = 5: N(i) = temp +1 => 1 coin

j = 1: temp = min(N(i – dj), temp) => min(N(4) , inf) => 2

j = 2: temp = min(N(i – dj), temp) => min(N(2) , 2) => 2

j = 3: temp = min(N(i – dj), temp) => min(N(0) , 2) => 0

i = 6: N(i) = temp +1 => 2 coins

j = 1: temp = min(N(i – dj), temp) => min(N(5) , inf) => 1

j = 2: temp = min(N(i – dj), temp) => min(N(3) , 1) => 1

j = 3: temp = min(N(i – dj), temp) => min(N(1) , 1) => 1

i = 7: N(i) = temp +1 => 3 coins

j = 1: temp = min(N(i – dj), temp) => min(N(6) , inf) => 2

j = 2: temp = min(N(i – dj), temp) => min(N(4) , 2) => 2

j = 3: temp = min(N(i – dj), temp) => min(N(2) , 2) => 2

i = 8: N(i) = temp +1 => 2 coins

j = 1: temp = min(N(i – dj), temp) => min(N(7) , inf) => 3

j = 2: temp = min(N(i – dj), temp) => min(N(5) , 3) => 1

j = 3: temp = min(N(i – dj), temp) => min(N(3) , 1) => 1

i = 9: N(i) = temp +1 => 3

j = 1: temp = min(N(i – dj), temp) => min(N(8) , inf) => 2

j = 2: temp = min(N(i – dj), temp) => min(N(6) , 2) => 2

j = 3: temp = min(N(i – dj), temp) => min(N(4) , 2) => 2

Answer: it will take 3 coins for 9 cents back

1. Coin-Collecting Problem
   1. In words, how is different from book?

With the addition of the inaccessible squares it adds an additional check that would need to be performed to see if the path can be taken in a way.

It also forces (a bit but not entirely) that you need to start at the start and go down in a bottom up approach rather than starting at the end and going backwards and terminating the sequence if no move is possible.

* 1. Adjust pseudo code to follow new rule

AdjustCoinCollect(C[1…n, 1…m])

F[1,1] = C[1,1]

For j = 2 to m //rows

If C[1,j] is inaccessible //check col

Break // don’t keep going down the cols

else

F[1,j] = F[1,j-1] + C[1,j]

For i = 2 to n //cols

If C[I, j] is inaccessible // check current

Break

else

F[i,1] = F[i-1,1] + C[i,1]

For j = 2 to m

If C[i,j+1] is inaccessible // check col

Break // don’t keep going down the cols

else

F[i,j] = max(F[i-1,j], F[i,j-1] ) + C[i,j]

Return F[n,m]

* 1. Use to solve example and fill out two grids

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X |  | o |  |  |
| o |  |  | X | o |  |
|  | o |  | X | o |  |
|  |  |  | o |  | o |
| X | X | X |  | o |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | X |  |  |  |  |
| o |  |  | X | o |  |
|  | o |  | X | o |  |
|  |  |  | o |  | o |
| X | X | X |  | o |  |

1. Knapsack (bottom-up DP)
2. Optimal Binary Search Trees
3. Warshall’s algo
4. Floyd’s algo
5. Edit Distance Problem
   1. Cange letter
   2. Insert letter
   3. Delete letter
6. Develop word distance
   1. Pseudo code
   2. Best/Worst cases
   3. Build or explain in detail