Anthony Huynh

CS 325

Homework 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length *i* | 1 | 2 | 3 | 4 |
| Price | 1 | 5 | 8 | 9 |
|  | 1 | 2.5 | 2.67 | 2.25 |

Consider a rod of length 4. Per the greedy strategy, we would cut the rod into lengths 3 and 1. The total price would be 8+1=9

However, if the rod were to be cut into two 2 pieces of length 2, then the total price would be

5+5=10, which is greater than 9.

1. MODIFIED-CUT-ROD(p, n, c){

int array[n];

array[0] = 0;

for (int i = 0; i<n; i++){

int q = p[i];

for (int j = 1; j< i-1; j++){

q = max(q, p[j] + r[i-j] – c);

}

r[j] = q;

}

return r[n];

}

* 1. The following code currently returns the fewest amount of coins needed to make an amount A using an array V that holds all denominations of change possible. It starts by making a change array that holds A elements. Each elements starts with being A+1 to show that it is empty. Each element checks the minimum used for the last eligible element based on all coin denominations by subtracting it(ex: with 1, 2, and 5 coins, element 5 will check elements 4, 3, and 0). It will now add 1 to that minimum, and compare to the current minimum. It loops through this cycle to check all denominations.

makeChange(V[], A){

m=0;

change[0] =0;

for i=1 to A

change[i]=A+1;

for j=0 to n-1

amountToCheck = i – V[j]

if amountToCheck >= 0

change[i] = min(change[amountToCheck] + 1, change[i]);

m = change[A]

return m

}

* 1. The running time would be O(A\*n) [A= amount, n= number of different coins]