Dynamic Programming Rod Cutting Problem

CSCI 3100

Rod Cutting Problem

Given a rod of length N inches and a list of prices for rods of smaller sizes (in inches), cut the rod into some number of pieces such that the value of the rod pieces is maximized

Input:

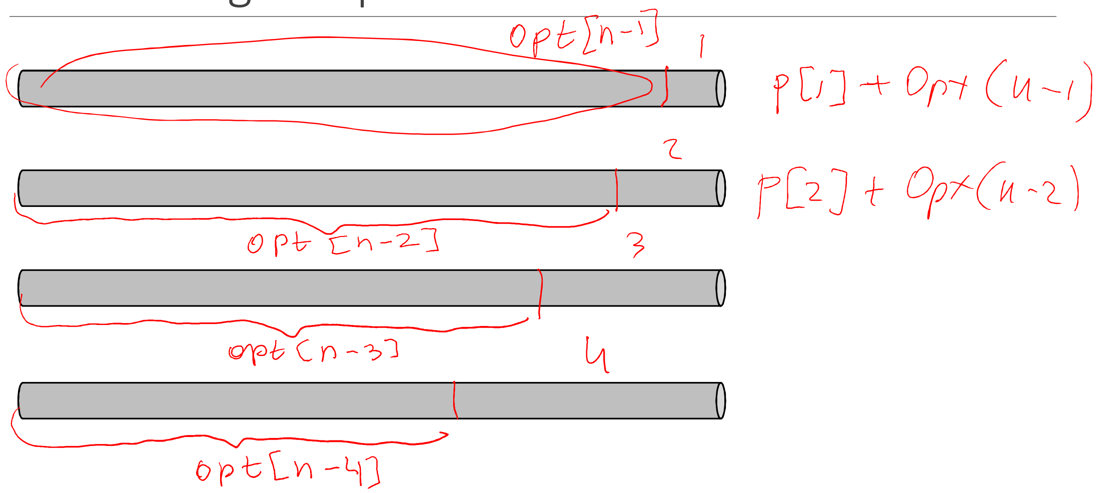
N – length of the rod

P = {p[1], p[2], … p[N]}, where p[i] is the price of the rod of length I

Output:

Opt[N] – maximum value of the rod pieces

Rod Cutting Sub-problems



Rod Cutting Overlapping Sub-problems

opt[n]

p[1]+opt[n-1]   p[2]+opt[n-2]   . . . . . .   p[n]+opt[0]

p[1]+opt[n-2]   p[2]+opt[n-3]

same problem

Rod Cutting Recurrence

Opt[j] the largest profit of cutting a rod of length j

Opt[j] = max(Opt[j-1] + p[1], Opt[j-2] + p[2], …, Opt[0] + p[j])

We'll compute and memorize Opt[0], Opt[1], … , Opt [N].

Opt[0] = 0

Opt[1] = p[1]

Opt[2] = max (Opt[1]+p[1], Opt[0]+p[2])

Rod Cutting Example

N = 4

X = {2, 3, 4, 5}

Solution

Opt	∅	p[1]+ Opt[0] 2	p[1]+ opt[1] 4	p[1]+ opt[2] 6	p[1]+ Opt[3] 8
	0	1	2	3	4

Opt[j] = max(Opt[j-1] + p[1], Opt[j-2] + p[2], …, Opt[0] + p[j])

Algorithm

```
BOTTOM-UP-CUT-ROD(p, n)
let r[0..n] be a new array
r[0] = 0
for j = 1 to n
    q = -∞
    for i = 1 to j
        q = max(q, p[i] + r[j - i])
    r[j] = q
return r[n]
```

$1 + 2 + 3 \; . \; . \; . \; . \; n =$

$$\sum_{i=1}^{n} i = \Theta(n^2)$$

Time complexity: $\Theta(n^2)$ Space complexity: $\Theta(n)$

Prove that the algorithm works for any N

Does the algorithm work for non-integer K

Does the algorithm work for non-integer w_j