Home work 2

Vasanth Reddy Baddam

02/13/2020

I pledge that this test/assignment has been completed in compliance with the Graduate Honor Code and that I have neither given nor received any unauthorized aid on this test/assignment
Name: Vasanth Reddy Baddam
Signature: VB

Q1.
p = [5,10,3,12,5,50,6]
n = length(p) - 1
Let's assume two matrices m and s of size n*n
Let's say we are going through the chain of matrices of length
for l = 1, Assuming that we only have one matrix on the multiplication. Then the cost for that multiplication would be zero.
So we replace the elements of matrix m ($m[i][i]$) with zero
for l = 2

$$m[1,2] = m[1,1] + m[2,2] + p[0] * p[1] * p[2]$$

$$m[1,2] = 0 + 0 + 5 * 10 * 3$$

$$m[1.2] = 150$$

$$m[2,3] = m[2,2] + m[3,3] + p[2] * p[3] * p[4]$$

$$m[2,3] = 0 + 0 + 10 * 3 * 12$$

$$m[2.3] = 360$$

$$m[3,4] = m[3,3] + m[4,4] + p[3] * p[4] * p[5]$$

$$m[3,4] = 0 + 0 + 3 * 12 * 5$$

$$m[3,4] = 180$$

$$m[4,5] = m[4,4] + m[5,5] + p[5] * p[4] * p[6]$$

$$m[4,5] = 0 + 0 + 12 * 5 * 50$$

$$m[4,5] = 3000$$

$$m[5,6] = m[5,5] + m[6,6] + p[5] * p[6] * p[7]$$

$$m[5,6] = 0 + 0 + 5 * 50 * 6$$

$$m[5,6] = 1500$$

for l = 2

$$m[1,3] = min(m[1,1] + m[2,3] + p[0] * p[1] * p[3]), (m[1,2] + m[3,3] + p[0] * p[2] * p[3])$$

$$m[1,3] = min(0 + 360 + 600), (150 + 0 + 180)$$

$$m[1,3] = min 960, 330$$

$$m[1,3] = 330$$

Using the recursive value from the above findings for $l = 2$ from $l = 1$
Similarly filling the matrix m, we get

i/j	1	2	3	4	5	6
1	0	150	330	405	1655	2010
2	0	0	360	330	2430	1950
3	0	0	0	180	930	1770
4	0	0	0	0	3000	1860
5	0	0	0	0	0	1500
6	0	0	0	0	0	0

K table is given by

i/j	1	2	3	4	5	6
1	0	1	2	2	4	2
2	0	0	2	2	2	2
3	0	0	0	3	4	4
4	0	0	0	0	4	4
5	0	0	0	0	0	5
6	0	0	0	0	0	0

Q2.

i/j	0	S	P	A	N	K	I	N	G
0	0	0	0	0	0	0	0	0	0
A	0	0	0	1	1	1	1	1	1
M	0	0	0	1	1	1	1	1	1
P	0	0	1	1	1	1	1	1	1
U	0	0	1	1	1	1	1	1	1
T	0	0	1	1	1	1	1	1	1
A	0	0	1	2	2	2	2	2	2
T	0	0	1	2	2	2	2	2	2
I	0	0	1	2	2	2	3	3	3
O	0	0	1	2	2	2	3	3	3
N	0	0	1	2	3	3	4	4	4

Direction table is given by:

i/j	0	S	P	A	N	K	I	N	G
0	0	0	0	0	0	0	0	0	0
A	0	↑	↑	↑	↖	←	←	←	←
M	0	↑	↑	↑	↑	↑	↑	↑	↑
P	0	↑	↖	↑	↑	↑	↑	↑	↑
U	0	↑	↑	↑	↑	↑	↑	↑	↑
T	0	↑	↑	↑	↑	↑	↑	↑	↑
A	0	↑	↑	↖	←	←	←	←	←
T	0	↑	↑	↑	↑	↑	↑	↑	↑
I	0	↑	↑	↑	↑	↑	↖	←	←
O	0	↑	↑	↑	↑	↑	↑	↑	↑
N	0	↑	↑	↑	↖	←	↑	↖	←

Following the diagonal arrows we get the longest common sub sequence is PAIN

Q4.

Given that the splits in quick sort are in the proportion of $1-\alpha$ and α
As we seen that minimum depth always take the smaller part of the split i.e.. α. Where as the maximum depth always takes the larger split i.e.. $1-\alpha$. As the number of passes increases, α and $1-\alpha$ changes to α^a and $1-\alpha^b$ respectively
At the end of the tree with one node remaining for minimum depth, then

$$n * \alpha^a = 1$$

$$\alpha^a = \frac{1}{n}$$

applying log on both sides,

$$a * \log(\alpha) = -\log(n)$$

$$a = -\frac{\log(n)}{(\alpha)}$$

At the end of the tree with one node remaining for maximum depth, then

$$n * (1-\alpha)^b$$

$$(1-\alpha)^b = \frac{1}{n}$$

applying log on both sides,

$$b * \log(1-\alpha) = -\log(n)$$

$$b = -\frac{\log(n)}{(1-\alpha)}$$

Q5.

As we know, the condition i.e ..

$$n = \frac{D * E}{P + E}$$

where D is Maximum number of elements can be in an array
E is Space for data value
P is Space for pointer

a).

space of an array = number of elements can hold*size of data field

$$\text{size of an array} = 20 * 8$$

bytes

$$\text{size of an array} = 160$$

bytes
space for an one node in linked list requires $= 8 + 4$ bytes
Condition to an linked list when:

$$n * 12 <= 160$$

$$n <= 13.33$$

for $n <= 13$, linked list needs space less than array

b).

space of an array = number of elements can hold*size of data field

$$\text{size of an array} = 30 * 2$$

bytes

$$\text{size of an array} = 60$$

bytes

space for an one node in linked list requires $= 4 + 2$ bytes

Condition to an linked list when:

$$n * 6 <= 60$$

for $n < 10$, linked list needs space less than array

c).

space of an array = number of elements can hold*size of data field

$$\text{size of an array} = 30 * 1$$

bytes

$$\text{size of an array} = 30$$

bytes

space for an one node in linked list requires $= 1 + 4$ bytes

Condition to an linked list when:

$$n * 5 <= 30$$

for $n < 6$, linked list needs space less than array

d).

space of an array = number of elements can hold*size of data field

$$\text{size of an array} = 40 * 32$$

bytes

$$\text{size of an array} = 1280$$

bytes

space for an one node in linked list requires $= 32 + 4$ bytes

Condition to an linked list when:

$$n * 36 <= 1280$$

$$n < 35.55$$

for $n < 35$, linked list needs space less than array

Q3.

Algorithm 1 Rod-Cutting w/Cost

```
procedure ROD-CUTTING(p,n,c)
  let r[0,1,..n] be a new array)
  r[0] = 0
  for j = 1 to n do
    q = p[j]
    for i = 1 to j -1  do
      q = max(q, p[i] + r[j-i] - c)
    r[j] = q
  return(r[n])
=0
```