Fundamental Algorithms, Home work - 4

Tulsi Jain

1. First Problem

Array, A =

19	2	11	14	7	17	4	3	5	15

Pivot is 19

Right array is empty

2	11	14	7	17	4	3	5	15		19

Pivot is 14

Replace pivot with first element

14	11	2	7	17	4	3	5	15		19

Pivot is 14

After Partitioning

11	2	7	4	3	5		14		17	15		19

Pivot is 2 for left array

Replace pivot with first element

2	11	7	4	3	5		14		17	15		19

Pivot is 2

After Partitioning

2		11	7	4	3	5		14		17	15		19

Pivot is 5

Replace pivot with first element

2		5	7	4	3	11		14		17	15		19

Pivot is 5

After Partitioning

2		4	3		5		7	11		14		17	15		19

Pivot is 4

After Partitioning

2		3		4		5		7	11		14		17	15		19

Pivot is 7

After Partitioning

2		3		4		5		7		11		14		17	15		19

Pivot is 15

Replace pivot with first element

2		3		4		5		7		11		14		15	17		19

Pivot is 15

After Partitioning

2		3		4		5		7		11		14		15		17		19

We got the, sorted array at the end.

2. Second Problem

Prove below function is maximum when either q = 0 or q = n 1

$$q^2 + (n - q - 1)2$$

Let's say an expression a $f(q)$ and find out $\frac{df}{dq}$

$$\frac{df}{dq} = 2q + 2(n - q - 1) * (-1)$$

To find maxima and minima we equate $\frac{df}{dq}$ to zero.

$$\frac{df}{dq} = 2q + 2(n - q - 1) * (-1) = 0$$

$$2q + 2q - 2n + 2 = 0$$

$$q = \frac{n-1}{2} \tag{2.1}$$

$$\frac{d^2 f}{dq^2} = 4$$

$q = \frac{n-1}{2}$ point would be the minima not the maxima as double derivative is always positive. This is parabolic function in q with positive sign on term q^2. It implies that maxmia would be at end points for a given range.

For q = 0;

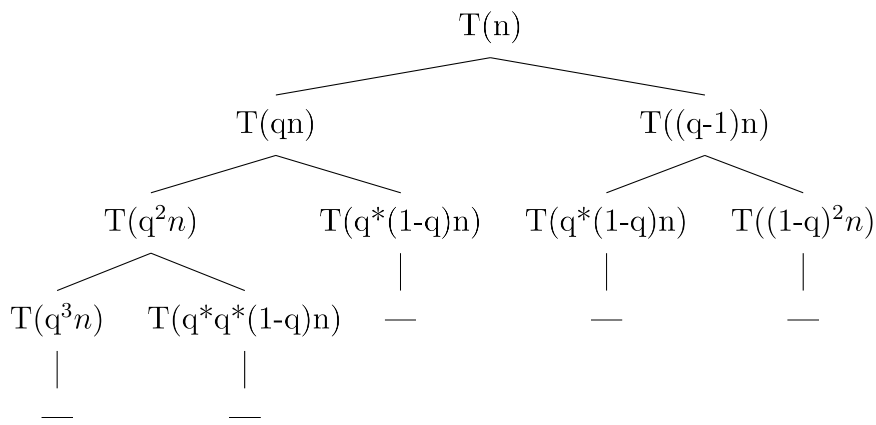
$$q^2 + (n - q - 1)2 = (n - 1)^2$$

For q = n - 1;

$$q^2 + (n - q - 1)2 = (n - 1)^2 + (n - n + 1 - 1)^2 = (n - 1)^2$$

Value at both end points are same. Hence, for a given range maxima would occur at at **q = 0 and q = n - 1**

3. Third Problem



All the branches are continuously to go downwards until reaches are end. Numbers of element are proportional to q^x and $(1-q)^x$. But as we move from left to right is $(1-q)^x$ is dominating over q^x. As $0 < alpha < \frac{1}{2}$ left branch would reach the end first. Hence, minimum height of the recursion is corresponding to leftmost subbranch.

$$\alpha^{h_{min}} * n \approx 1$$

$$h_{min} \log \alpha \approx \log \frac{1}{n}$$

$$h_{min} \log \alpha \approx -\log n)$$

$$h_{min} \approx -\frac{\log n}{\log \alpha}$$

Maximum height of the recursion would be corresponding to rightmost subbranch.

$$(1-\alpha)^{h_{max}} * n \approx 1$$

$$h_{max} \log (1-\alpha) \approx \log \frac{1}{n}$$

$$h_{max} \log (1-\alpha) \approx -\log n)$$

$$h_{max} \approx -\frac{\log n}{\log (1-\alpha)}$$

4. Fourth Problem

Let's consider an array of size n, which element from { a_1, a_2, a_3 .... a_n}. Let's pick the *ith* maximum element as a pivot. It would divide the array into (n - m):(n - m -1) ratio.
As pivot selection is random, any th maximum number can be selected.
If selected element is first maxi-mun then distribution would be **n - 1 : 0**
If selected element is first maxi-mun then distribution would be **n - 2 : 1**
.....
At some point it distribution would be **n - 1 -** α**n :** α**n** eq. 1
.....
further down the lines it would be α**n : n - 1 -** α**n** eq. 2
.....
At the end it would be **0 : n - 1**
Each of the above distribution is equally likely as it is given that we are picking the pivot randomly.
Randomly selected pivot would give more balanced distribution between equation 1 and 2 than αn. In other words, If pivot is in range $[\alpha n + 1, n - \alpha n - 2]$th maximum element.
Let's find out its probability.
Count of favorable position is
$n - \alpha n - 2 - \alpha n - 1$
$= n - 2\alpha n - 3$
Probability is $\frac{n-2\alpha n-2}{n}$
= 1 - 2α - $\frac{3}{n}$
$\approx$ 1 - 2α (for sufficienty large value of n)

5. Fivth Problem

For Quick Sort best running time
Let's pivot, p divides array of lenght, n into r and n - 1 - q elements. Hence q lies between 0 and n - 1.

$$T(n) = min(T(q) + T(n - q - 1)) + \Theta(n)$$

Induction Hypothesis

$$T(k) >= ck \log k \quad 1 < k < n$$

Base Case, when k = 2

$$T(2) >= c2 \log 2$$

This is valid as as partitioning would take k = 2 and number of time is $\log 2$
Let's check proof for n

$$T(g) <= c(min(q \log q + (n - q - 1) \log (n - q - 1))) + \Theta(n)$$

We need to find out minimum of $E = q \log q + (n - q - 1) \log (n - q - 1)$.

$$\frac{dE}{dq} = 1 + \log q + (-1) * \frac{n - q - 1}{n - q - 1} + (-1) * \log (n - q - 1))$$

$$\frac{dE}{dq} = 1 + \log q + -1 - \log (n - q - 1))$$

Equate $\frac{dE}{dq}$ to 0.

$$1 + \log q + -1 - \log (n - q - 1)) = 0$$

$$q = n - q - 1$$

$$q = \frac{n - 1}{2}$$

Only single stationary point

Let's find $\frac{d^2E}{dq^2}$

$$\frac{d^2E}{dq^2} = \frac{1}{q} + \frac{1}{n - q - 1}$$

put $q = \frac{n-1}{2}$, then $\frac{d^2E}{dq^2} = \frac{4}{n-1}$. $\frac{d^2E}{dq^2}$ positive implies this is point of minima.
Hence put, $q = \frac{n-1}{2}$

$$T(n) >= c\frac{n - 1}{2} * \log \frac{n - 1}{2} + c\frac{n - 1}{2} * \log \frac{n - 1}{2} + \Theta(n)$$

$$T(n) >= c(n-1) * \log \frac{n-1}{2} + \Theta(n)$$

$$T(n) >= c(n-1) * (\log (n-1) - \log 2) + \Theta(n)$$

$$T(n) >= cnlog(n-1) - cn \log 2 - c \log n - 1 + c \log 2 + \Theta(n)$$

$$T(n) >= cnlog\frac{n}{2} - (cn \log 2 + c \log n - 1 - c \log 2 - \Theta(n))$$

$$T(n) >= cnlogn - (cn - cn \log 2 + c \log n - 1 - c \log 2 - \Theta(n))$$

Second Part in open parenthesis should be non postitive $cn - cn \log 2 + c \log n - 1 - c \log 2 <= \Theta(n)$. This is perfectly valid as both side are linear in n and we can chosse constant c such that this always holds true. First part is equal to Induction hypothesis.
Hence best running time in quick sort is $\Omega(n \log n)$