

HW4

1. From Ryan Duffy.

- 1) Illustrate the operation of PARTITION on the array $A = \{13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11\}$.

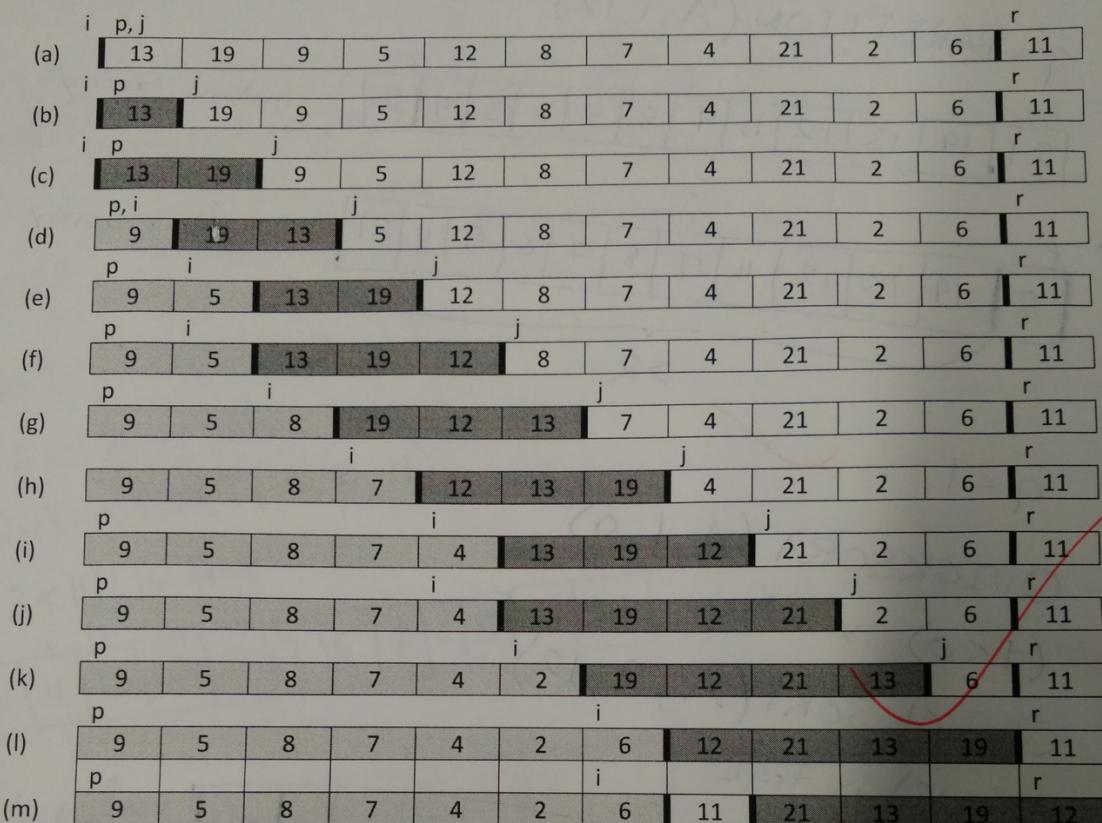


Figure 1. An illustration of the operation of PARTITION on array A.

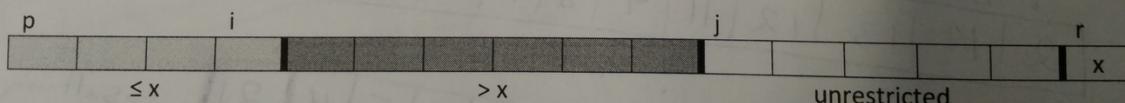


Figure 2. The four regions maintained by the procedure PARTITION on the subarray $A[p..r]$.

QUICKSORT(A, p, r)

1. if $p < r$
 2. $q = \text{PARTITION}(A, p, r)$
 3. $\text{QUICKSORT}(A, p, q - 1)$
 4. $\text{QUICKSORT}(A, q + 1, r)$

PARTITION(A, p, r)

1. $x = A[r]$
 2. $i = p - 1$
 3. for $j = p$ to $r - 1$
 4. if $A[j] \leq x$
 5. $i = i + 1$
 6. exchange $A[i]$ with $A[j]$
 7. exchange $A[i + 1]$ with $A[r]$
 8. return $i + 1$

2. From Fanyou Meng.

2. QuickSort Algorithm

Descending Order:

Assume $A = \{8, 6, 5, 2, 1, 0\}$

(a)	i	P	j	r
	8	6	5	2 1 0

(b)	i	P	j	r
	8	6	5	2 1 0

(c)	i	P	j	r
	8	6	5	2 1 0

(d)	i	P	j	r
	0	6	5	2 1 8

- $q = i+1 = P$, call recursive function
QuickSort($A, P, q-1$) & QuickSort($A, q+1, r$).

Since only the second recursive function will actually work, ($P > q-1$)
draw the diagrams of the 2nd part here:

(e)	i	P	j	r
	0	6	5	2 1 8

(f)	i	P	j	r
	0	6	5	2 1 8

(g)	i	P	j	r
	0	6	5	2 1 8

- $q = i+1 = r$

Call recursive function QuickSort($A, P, q-1$) & QuickSort($A, q+1, r$)

draw diagrams of QuickSort($A, P, q-1$)

(h)	i	P	j	r
	0	6	5	2 1 8

(i)	i	P	j	r
	0	6	5	2 1 8

(j)	i	P	j	r
-----	---	---	---	---

(k)	i	P	j	r
-----	---	---	---	---

- $q = i+1 = P$, doesn't work

Call QuickSort($A, P, q-1$) & QuickSort($A, q+1, r$)

draw diagrams of Quicksort($A, q+1, r$)

(L) $\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 5 & 2 & 6 & 8 \\ \hline \end{array}$

(M) $\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 5 & 2 & 6 & 8 \\ \hline \end{array}$

(N)

$\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 5 & 2 & 6 & 8 \\ \hline \end{array}$

$q = i+1 = r$

doesn't work

- Call Quicksort($A, p, q-1$) & Quicksort($A, q+1, r$)

draw diagrams of Quicksort($A, p, q-1$)

(O) $\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 5 & 2 & 6 & 8 \\ \hline \end{array}$

(P) $\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 5 & 2 & 6 & 8 \\ \hline \end{array}$

Reordered but took \rightarrow A

(Q) $\begin{array}{|c|c|c|c|c|} \hline i & p & j & r \\ \hline 0 & 1 & 2 & 5 & 6 & 8 \\ \hline \end{array}$

$q = i+1 = p$

doesn't work

- Call Quicksort($A, p, q-1$) & Quicksort($A, q+1, r$)

• Reach the bottom of recursion.

Array A is $\boxed{0 \ 1 \ 2 \ 5 \ 6 \ 8}$ now.

Therefore, if the array is in descending order, q is either p or r .

When the pivot is the current smallest element in the subarray,

q is equal to p ; when the pivot is the largest element in the current subarray, q is equal to r .

Quicksort Algorithm Running time

The PRACTITIONER

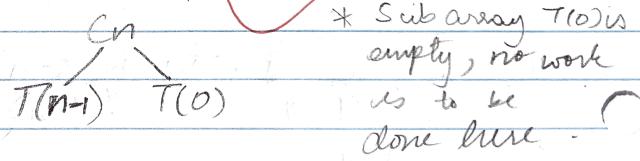
3. From Chahat Gupta

Q.3 Quicksort (A, P, R)

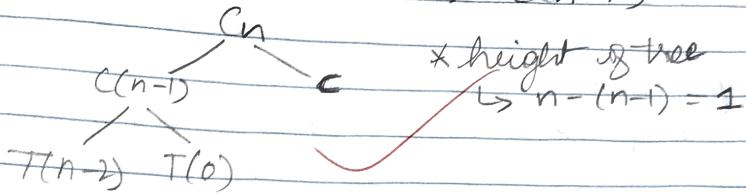
		C_i	# of times
1	if $P < R$	C_1	1
2	$q = \text{PARTITION}(A, P, R)$	$\Theta(n)$	1
3	Quicksort ($A, P, q-1$)	$T(0)$	1
4	Quicksort ($A, q+1, R$)	$T(n-1)$	1

$$\Rightarrow T(n) = T(n-1) + T(0) + \Theta(n) + c$$

$$= T(n-1) + T(0) + cn$$

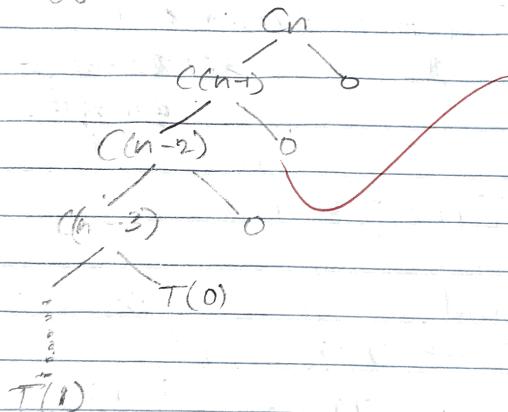


$$T(n-1) = T(n-2) + T(0) + c(n-1)$$



$$T(n) = c_n + c(n-1) + c(n-2) + \dots + c$$

$$= \sum_{i=0}^n (c(n-i))$$



$$T(n) = c(n + (n-1) + (n-2) + \dots + 1) = cn(n+1)/2 = \Theta(n^2)$$

4. From Guangxin Ye.

$$\text{upper bound } T(n) \leq T(n-1) + cn \quad T(n) = O(n^2)$$

$$\text{Guess } T(n) \leq dn^2 \quad (c, d \text{ are positive constant})$$

$$T(n-1) \leq d(n-1)^2 = dn^2 - 2dn + d$$

$$T(n) \leq dn^2 - 2dn + d + cn = dn^2 + (c-2d)n + d$$

$$\Rightarrow T(n) \leq dn^2 \text{ if } c-2d < 0 \Rightarrow c < 2d$$

$$\therefore T(n) = O(n^2)$$

lower bound

$$T(n) \geq T(n-1) + cn \quad T(n) = \Omega(n^2)$$

Guess

$$T(n) \geq dn^2$$

$$T(n-1) \geq d(n-1)^2 = dn^2 - 2dn + d$$

$$T(n) \geq dn^2 - 2dn + d + cn = dn^2 + (c-2d)n + d$$

$$\Rightarrow T(n) \geq dn^2 \text{ if } c-2d \geq 0 \Rightarrow c \geq 2d$$

$$\therefore T(n) = \Omega(n^2)$$

$$\text{overall } T(n) = \Theta(n^2)$$

5. From Ishita panda.

Pg 4

5) There is 3:7 ratio split
 $\therefore T(n) = T\left(\frac{3n}{10}\right) + T\left(\frac{7n}{10}\right) + cn$

(a) $T(n)$

(b)

(c)

Unbalanced tree, where each level has a height of n .
 ht. of shorter tree : ~~$\log_{10/3} n$~~ $\log_{10/3} n$
 taller : $\log_{10/7} n$

$T(n) \geq C \sum_{i=0}^{\log_{10/3} n} n = \Omega(n \lg n)$

$T(n) \leq C \sum_{i=0}^{\log_{10/7} n} n = O(n \lg n)$

$\therefore T(n) = \Theta(n \lg n)$

May use Theorem: $T(n) = T(\alpha n) + T((1-\alpha)n) + \Theta(n)$, where $0 < \alpha \leq 1/2$
 $T(n) = \Theta(n \lg n)$

6. From Third Hsu Myat Aung

6)-
7.2 - 4

Answer: The problem of converting time of transaction ordering to check number ordering is the problem of sorting almost sorted input.

1). The insertion sort takes $\Theta(n)$ time to sort an input that is already sorted.

the insertion sort will take $\Theta(n)$ to sort an input that is almost sorted.

2). Quicksort takes $\Theta(n^2)$ time to sort an input that is already sorted.

the Quicksort will take $\Theta(n^2)$ to sort an input that is almost sorted.

Performance of insertion sort is better than quick sort by considering the running times that for the problem of converting time of transaction ordering to check number ordering.

7. From Menaka Gnanavel.

Split $(1-\alpha)$ to α ($0 \leq \alpha \leq \frac{1}{2}$).

$\Rightarrow T(n) = T(\alpha n) + T((1-\alpha)n) + O(n).$

MINIMUM HEIGHT :-

$$\log_{1/\alpha} n = \frac{\log_b n}{\log_b 1/\alpha} = -\frac{\log_b n}{\log_b \alpha}$$

$$= -\frac{\log n}{\log \alpha}$$

MAXIMUM HEIGHT :-

~~$$\log_{1/(1-\alpha)} n = \frac{\log_b n}{\log_b 1/(1-\alpha)} = -\frac{\log_b n}{\log_b (1-\alpha)}$$~~

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