CS & IT ENGINEERING

Algorithms

Design Strategies

Lecture No.- 04

By- Dr. Khaleel Khan Sir

Recap of Previous Lecture

Topic Merge Sort

Topic Quick Sort

Topic

Topic

Topic

Topics to be Covered

Topic	Matrix Multiplication
Topic	Master Method
Topic	
Topic	
Topic	

5. Let P be a quick sort program to sort numbers in ascending order. $n=4$
Let t_1 and t_2 be the time taken by the program for the inputs [1 2 3 4] and [5 4 3 2 1], respectively. Which of the following holds?
$n=5$

(a) $t_1 = t_2$
(b) $t_1 > t$
(c) $t_1 < t_2$
(d) $t_1 = t_2 = 5 \log 5$

6. Let P be a Quick Sort Program to sort numbers in ascending order suing the first element as pivot. Let t1 and t2 be the number of comparisons made by P for the inputs {1, 2, 3, 4, 5} and {4, 1, 5, 3, 2} respectively. Which one of the following holds?
W.C (n^2) B.C $(n \log n)$

(a) $t_1 = 5$
(b) $t_1 < t_2$
(c) $t_1 > t_2$
(d) $t_1 = t_2$

7. Quick-sort is run on two inputs shown below to sort in ascending order taking first element as pivot

i. 1, 2, 3....n

ii. n, n – 1, n – 2,...,2,1

Let C_1 and C_2 be the number of comparisons made for the inputs (i) and (ii) respectively. Then,

(a) $C_1 < C_2$

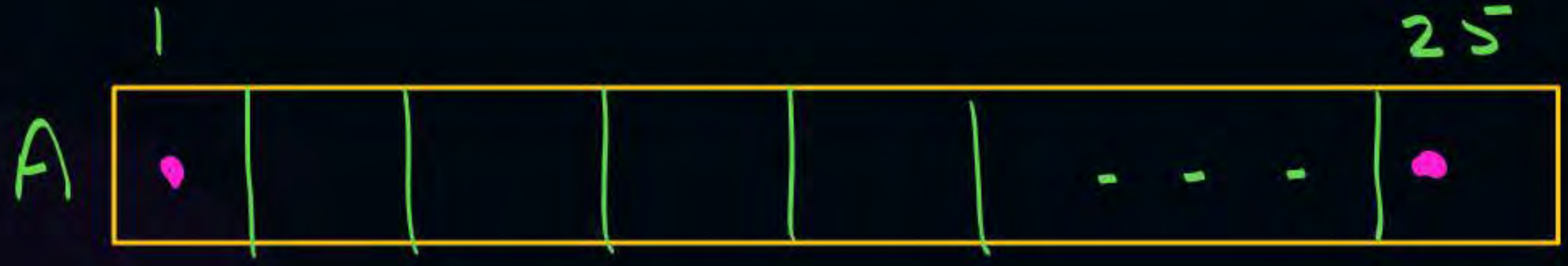
(b) $C_1 > C_2$

(c) $C_1 = C_2$

(d) We cannot say anything for arbitrary n

8. An array of 25 distinct elements is to be sorted using quicksort. Assume that the pivot element is chosen uniformly at random. The probability that the pivot element gets placed in the worst possible location in the first round of partitioning (rounded off to 2 decimal places) is 0.08.

(NAT)



$$\frac{1}{25} + \frac{1}{25} = 2/25 = 0.08$$

5) Matrix Multiplication :

$A_{n \times n}$; $B_{n \times n}$; $C_{n \times n}$

1) $A \pm B = C$

$O(n^2)$

```
for i <- 1 to n
    for j <- 1 to n
        C[i,j] = A[i,j] ± B[i,j]
```

$$A \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}_{2\times 2} * B \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}_{2\times 2}$$

$$= C \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

$c_{11} = a_{11} * b_{11} + a_{12} * b_{21}$

$c_{12} = a_{11} * b_{12} + a_{12} * b_{22}$

$c_{21} = a_{21} * b_{11} + a_{22} * b_{21}$

$c_{22} = a_{21} * b_{12} + a_{22} * b_{22}$

2) $A * B = C$ [School Method / Non-DC]

```
for i <- 1 to n
  for j <- 1 to n
    c[i,j] = 0;
    for K <- 1 to n
      c[i,k] += A[i,k] * B[k,j]
```

$O(n^3)$

Can we multiply 2-Square Matrices of order $n \times n$, using D and C Method?

$$A = \begin{array}{cc} A_{11} & A_{12} \\ \left[\begin{array}{cc|cc} 1 & 2 & 3 & 4 \\ 5 & 6 & 8 & 2 \\ \hline 1 & 3 & 5 & 7 \\ 9 & 1 & 2 & 5 \end{array}\right]_{4\times 4} \\ A_{21} & A_{22} \end{array} \quad B = \begin{array}{cc} B_{11} & B_{12} \\ \left[\begin{array}{cc|cc} 5 & 6 & 7 & 2 \\ 9 & 8 & 1 & 9 \\ \hline 6 & 5 & 4 & 2 \\ 3 & 1 & 5 & 6 \end{array}\right]_{4\times 4} \\ B_{21} & B_{22} \end{array} = C \left[\begin{array}{c|c} C_{11} & C_{12} \\ \hline C_{21} & C_{22} \end{array}\right]_{4\times 4}$$

Sub-Matrix multip.

Sub Matrix Add (n^2)

$n/2$

$$C_{11} = (\underbrace{A_{11} * B_{11}}_{T(n/2)}) + (\underbrace{A_{12} * B_{21}}_{T(n/2)}) \quad -1$$

$n/2$

$$C_{12} = A_{11} * B_{12} + A_{12} * B_{22} \quad -2$$

$$C_{21} = A_{21} * B_{11} + A_{22} * B_{21} \quad -3$$

$$C_{22} = A_{21} * B_{12} + A_{22} * B_{22} \quad -4$$

Multipl.
+
Additions

(Conquer)

16×16
↓
8×8
↓
4×4
↓
$\boxed{2 \times 2}$

1×1

→ Let $T(n)$ repr. Time Complexity to multiply Two Square Matrices A & B of order $n \times n$;

$$T(n) = C \qquad n \le 2$$

$$O(n^{\log_2 8}) \qquad = 8 \cdot T(n/2) + bn^2 \quad , n > 2 \; , b > 0$$

$$T(n) = 8T(n/2) + bn^2 \; - (1)$$

$$T(n/2) = 8T(n/4) + bn^2/4 \; - (2)$$

$$T(n) = 8\left[8T(n/4) + bn^2/4\right] + bn^2$$

$$= 64 \cdot T(n/4) + 3bn^2 \; - (3)$$

$$= 8^2 \cdot T(n/2^2) + (2^2 - 1)bn^2 \; - (4)$$

$$= 8^k \cdot T(n/2^k) + (2^k - 1)bn^2 \; - (5)$$

$$\frac{n}{2^k} = 1 \Rightarrow n = 2^k \Rightarrow k = \log_2 n$$

$$T(n) = 8^{\log_2 n} c + (n-1)bn^2$$

$$= n^3 \cdot c + bn^3 - bn^2 \; - (6)$$

$$= cn^3 + bn^3 - bn^2 \Rightarrow O(n^3)$$

T.C using DandC-Method

$$= O(n^3)$$

→ In D and C, there are presently 8 - Sub Matrix Multiplications Involved in Eq's $c_{11} \cdots c_{22}$;

→ Time-Complexity will get reduced, only if the no. of Sub-matrix Multiplications are reduced from 8 to a lesser value;

"STRASSEN"

↳ research

$a * b$

$[a+a+a+a \cdots + a]$

Topic : STRASSEN'S MATRIX MULTIPLICATION

$P = (A_{11} + A_{22})(B_{11} + B_{22})$

$Q = (A_{21} + A_{22})B_{11}$

$R = A_{11}(B_{12} - B_{22})$

$S = A_{22}(B_{21} - B_{11})$

$T = (A_{11} + A_{12})B_{22}$

$U = (A_{21} - A_{11})(B_{11} + B_{12})$

$V = (A_{12} - A_{22})(B_{21} + B_{22})$

$$\left.\begin{aligned} C_{11} &= P + S - T + V \\ C_{12} &= R + T \\ C_{21} &= Q + S \\ C_{22} &= P + R - Q + U \end{aligned}\right\}$$

→ $A, B, C : n \times n$

→ $A_{ij} ; B_{ij} ; C_{ij} : \frac{n}{2} \times \frac{n}{2}$

→ $P, Q, R, S, T, U, V : \frac{n}{2} \times \frac{n}{2}$ (Additional Matrices)

→ $T(n)$ repr. T.C of STRAS_DANDC (n×n)

$$T(n) = \begin{cases} C, & n \le 2 \\ 7 \cdot T(n/2) + bn^2, & n > 2 \end{cases}$$

$$T(n) = 7 \cdot T(n/2) + bn^2 \quad \text{— (1)}$$

$$T(n/2) = 7T(n/4) + bn^2/4 \quad \text{— (2)}$$

$$T(n) = 7\left(7 \cdot T(n/4) + bn^2/4\right) + bn^2 \quad \text{— (3)}$$

$$= 49 \cdot T(n/4) + \left(\frac{7}{4}\right)^1 bn^2 + \left(\frac{7}{4}\right)^0 bn^2 \quad \text{— (4)}$$

$$= 7^2 \cdot T(n/2^2) + bn^2 \cdot \sum_{i=0}^{1} (7/4)^i$$

$$\vdots$$

$$= 7^K \cdot T(n/2^K) + bn^2 \cdot \sum_{i=0}^{K-1} (7/4)^i$$

$$S_n = \frac{a(r^n - 1)}{r - 1} = \frac{1\left((7/4)^K - 1\right)}{\left(\frac{7}{4} - 1\right)}$$

$$\sum_{i=1}^{n} x^i < x^{n+1} \qquad \underline{\underline{x > 1}}$$

$$\sum_{i=1}^{3} 2^i < 2^4$$

$$T(n) < 7^K \cdot c + bn^2 \cdot \left(\frac{7}{4}\right)^K$$

$$< c \cdot 7^K + b \cdot \cancel{n^2} \cdot \frac{7^K}{\cancel{4^K}}$$

$$\frac{n}{2^K} = 1$$

$$n = 2^K$$

$$K = \log n$$

$$T(n) < d \cdot 7^K < d \cdot 7^{\log n} \Rightarrow n^{\log_2 7}$$

$$\boxed{T(n) < n^{2.81}}$$

Space Complexity :

1) School Method : $O(1)$

2) D and C - Method : $O(\log n)$

3) Strassen's Method : $\log n + n^2$

$= O(n^2)$

Master Theorem (Method) for Solving DandC Recurrences;

$$T(n) = a \cdot T(n/b) + f(n), \quad n > d \quad ; \quad \boxed{a \geq 1;\ b > 1;\ f(n) \text{ is } +ve}$$
$$= C \qquad\qquad , \quad n \leq d$$

Solving DandC Recurrences

- Back Substitution
  - Value
  - Order
- Basic Maths, Ability
- Master Theorem → $\dfrac{\text{Order}}{\text{Theta}}$
- (Recursion Tree)

Max Min

$$T(n) = 2T(n/2) + 2 = \left(\frac{3n}{2} - 2\right)$$ only with Back Substitution

evaluate ?

$\hookrightarrow O(n)$: Mas. Method

Master Theorem:

$T(n) = a \cdot T(n/b) + f(n)$; $a \geq 1$; $b > 1$; $f(n)$: +ve

Case I: If $f(n)$ is $O(n^{\log_b a - \epsilon})$ for some $\boxed{\epsilon > 0}$, then

$$\boxed{T(n) \text{ in } \Theta(n^{\log_b a})}$$

Case II: If $f(n)$ is $\Theta(n^{\log_b a} * \log^K n)$ for some K, such that

a) $K \geq 0$, then $\boxed{T(n) \text{ in } \Theta(n^{\log_b a} * \log^{K+1} n)}$

b) $K = -1$, then $\boxed{T(n) \text{ in } \Theta(n^{\log_b a} * \log\log n)}$

Case III: If $f(n)$ is $\Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$, and

$\boxed{a \cdot f(n/b) \leq \delta \cdot f(n)}$ for some $\delta < 1$, then

$$\boxed{T(n) \text{ in } \Theta(f(n))}$$

① $T(n) = 4 \cdot T(n/2) + n$ -

$$\left.\begin{aligned} a &= 4 \\ b &= 2 \\ f(n) &= n \end{aligned}\right\} \log_b a = \log_2 4 = 2$$

Case I: n is it $O(n^{2-\epsilon})$ $\epsilon = 1$

$\epsilon = 0.5$

$n = O(n)$ ✓

$$\therefore \boxed{T(n) \text{ is } \Theta(n^2)}$$ ✓

2) $T(n) = 2 \cdot T(n/2) + n \cdot \log n$

$a = 2$

$b = 2$

$f(n) = n \cdot \log n$

$\log_2 2 = 1$

Case I : $n \cdot \log n$ is it $O(n^{1-\epsilon})$ ✗

Case II : $n \cdot \log n$ is it $\Theta(n^1 \cdot \log^k n)$ $K = 1$ a: ✓

$$\boxed{\therefore T(n) \text{ is } \Theta(n \cdot \log^2 n)}$$

3) $T(n) = T(n/3) + n$ ✓

$n \neq \Theta(\log^k n)$

$a = 1;$
$b = 3;$
$f(n) = n$

$\log_b a = \log_3 1 = 0$

Case 1: n is it $O(n^{0-\epsilon})$ ✗

Case 2: n is it $\Theta(n^0 \cdot \log^k n)$ ✗

Case 3: n is it $\Omega(n^{0+\epsilon})$ $\epsilon = 1$, $\epsilon = 0.5$ ✓

$a \cdot f(n/b) \leq \delta \cdot f(n)$ for $\delta < 1$

$\boxed{1 \cdot \frac{n}{3} \leq \delta \cdot n}$ $\delta = 1/3 < 1$

$\boxed{\therefore T(n) = \Theta(n)}$

4) $T(n) = 9 \cdot T(n/3) + n^{2.5}$

$a = 9$; $b = 3$; $f(n) = n^{2.5}$ $= f(n/3) = (n/3)^{2.5} = \left(\frac{n^{2.5}}{9\sqrt{3}}\right)$

$\log_3 9 = 2$

Case 1 : $n^{2.5}$ is it $O(n^{2-\epsilon})$ ✗

Case 2 : $n^{2.5}$ is it $\Theta(\underline{n^2 \cdot \log^k n})$ ✗

$(\underline{n^2\sqrt{n}})$

Case 3 : $n^{2.5}$ is it $\Omega(n^{2+\epsilon})$ $\epsilon = 0.5$ ✓

$a \cdot f(n/b) \le \delta \cdot f(n)$

$\boxed{\cancel{9} \cdot \frac{n^{2.5}}{\cancel{9}\sqrt{3}} \le \delta \cdot n^{2.5}}$ for $\delta = \frac{1}{\sqrt{3}} < 1$

$\boxed{\therefore T(n) = \Theta(n^{2.5})}$

① Max-Min :

$\hookrightarrow T(n) = 2T(n/2) + 2 \rightarrow \left(\frac{3n}{2} - 2\right) = O(n)$

$a = 2;\ b = 2;\ f(n) = C$

Case I: C is it $O(n^{1-\epsilon})$ $\epsilon = 1$ ✓

$\therefore T(n)$ is $\Theta(n^1)$

② Merge Sort:

$T(n) = 2 \cdot T(n/2) + n$ | $a = 2;\ b = 2$ $f(n) = n$

$\log_2 2 = 1$

Case I: n is it $O(n^{1-\epsilon})$ $\epsilon > 0$ ✗

Case II: n is it $\Theta(n \cdot \log^k n)$ $K = 0$ ✓

$\boxed{\text{a) } T(n) \text{ in } \Theta(n \cdot \log n)}$

3) Matrix Multipl.

a) D and C : $T(n) = 8T(n/2) + n^2$

Case I: n^2 is it $O(n^{3-\epsilon})$ $\epsilon = 1$

$$\boxed{\therefore T(n) = \Theta(n^3)}$$

b) Strassen's : $T(n) = 7 \cdot T(n/2) + n^2$

$\log_2 7 = 2.81$

Case 1: n^2 is it $O(n^{2.81-\epsilon})$ $\epsilon = 0.81$

$$\boxed{T(n) = \Theta(n^{2.81})}$$

4) Binary Search : $T(n) = T(n/2) + C$

$a = 1$; $b = 2$; $f(n) = C$; $\log_2 1 = 0$

Case I : C is it $O(n^{0-\epsilon})$ ✗

Case II : C is it $\Theta(n^0 \cdot \log^k n)$ $K = 0$

$\boxed{\text{a) } T(n) \text{ is } \Theta(\log n)}$

H/W :

1) $T(n) = 3T(n/2) + n$

2) $T(n) = 16 \cdot T(n/4) + n$

3) $T(n) = 4T(n/2) + \log n$

4) $T(n) = \sqrt{2} \cdot T(n/2) + \log n$

5) $T(n) = 6 \cdot T(n/3) + n^2 \cdot \log n$

6) $T(n) = 2 \cdot T(n/2) + \dfrac{n}{\log n}$

7) $T(n) = 4T(n/2) + n^2$

8) $T(n) = 2T(n/2) + \sqrt{n}$

9) $T(n) = 3T(n/3) + n$

10) $T(n) = 2^n \cdot T(n/4) + n$

11) $T(n) = 2 \cdot T(\sqrt{n}) + \log n$

THANK - YOU