Page:
Date: / /

Distributed & Parallel Algorithms.

* Algorithms :- finite set of steps to solve a specific problem. Basic properties are :- I/P (0/more), O/P (atleast 1), follow finiteness rule, effectiveness, must give clear O/P (unambiguous).

* Program :- Implementation of algorithm by using any programming language.

* Execution of program depends on system (based on processor and memory), s.t., runtime would be affected. But, designing of algorithm does not depend upon the platform (system independent) and hence, we find the space & time complexity manually.

* Parallel processing is used to solve a very large problem which might take days to execute on a single ~~problem~~ processor. so in this case we divide the problem into subproblems and allow each subproblem to execute on one processor and hence, the time complexity decreases.

* For DS, systems (message Passing) are located at remote located (loosely coupled systems), while PS, systems are located at same located (tightly coupled system).
(~~one global memory~~)
(shared memory / message Passing)

eg :- leader selections,
multi-core systems.

Page :
Date : / /

* Flynn's taxonomy :- to classify the different computer architectures.

(i) SISD (Single Instruction Single Data) :- (No parallelism) to execute single instruction for that single data. eg.:- $a+b$

(ii) SIMD (Single Instruction Multiple Data) :- eg:- $a+b$, $c+d$, $e+f$

(iii) MISD (Multiple Instruction Single Data) :- eg.:- $a+b$, $a-b$, $a*b$, a/b.

(iv) MIMD (Multiple Instruction Multiple Data) :- (Parallelism) eg.:- $+ \rightarrow a+b$ | $- \rightarrow e-f$
$* \rightarrow c*d$ | $/ \rightarrow g/h$.
all instructions & all data are different.

* Speedup :- for parallel algorithms.

$$S = \frac{T(n)}{T(n,p)}$$

Total work done $= p \cdot T(n,p)$

$$\text{Efficiency} = \frac{\text{Speed up}}{\text{No of processors used}}$$

$$= \frac{T(n)}{p \cdot T(n,p)}$$

Since, for single processor, n input $= T(n)$

~~for p processor~~ $\rightarrow T(n,p) = O(n^2)$

$O(n \log n)$

Page:
Date: / /

* Amdohl's law:-

The fraction of the problem that we cannot parallelize be f.

$$\text{speedup} = \frac{1}{f + \frac{1-f}{p}}$$

Let, $f = 0.5$, $p = 10$

$$\therefore S = \frac{1}{0.5 + \frac{0.5}{10}} = \frac{1}{0.55} \approx 1.98 \approx 2$$

Let, $f = 0.1$, $p = 10$

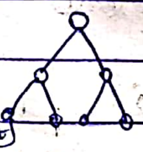
$$\therefore S = \frac{1}{0.1 + 0.09} = \frac{1}{0.19} \approx 5$$

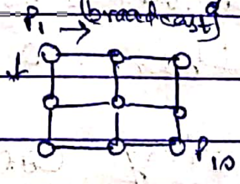
Hence, $\boxed{S \propto \frac{1}{f}}$

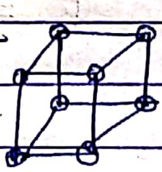
* Abstract Models of Parallel Computing :-

(i) RAM Model - For Intraprocessor operations:

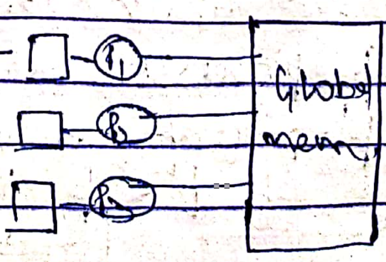
(i) Linear :- 

(ii) Binary tree :- 

(iii) Mesh :- 

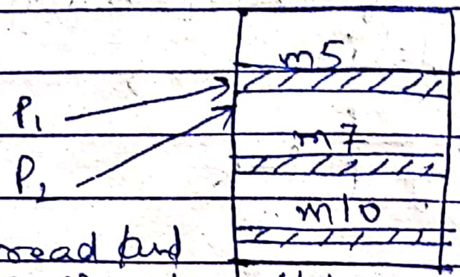
(iv) Hypercubes :- 

fixed conn n/w

→ Shared Memory :- 

eg :- PRAM model.

* PRAM model :— (Shared mem. location)



(can't read and write simultaneously)

(i) EREW :— Exclusive Read & Exclusive Write
(ii) CREW :— Concurrent Read & Exclusive Write
(iii) CRCW :— Concurrent Read & Concurrent Write

(to deal with concurrent write)

Priority CRCW — Arbitrary CRCW — Common CRCW

→ A[1...n]
(n bits)

A[0] = A[1] | A[2] | A[3] | ..... | A[n]

eg:- 0110100

for p-processors:
 Processor i (in parallel 1 ≤ i ≤ n) do:
  if (A[i] == 1) then A[0] = A[i];

A[0] → O/P using Common CRCW

O(1)

* Slow down lemma :—

p - processor machine → T (for any parallel algo).
p' - processor machine → T'
if p' < p then (slow down)

$\boxed{T' = \frac{pT}{p'}}$ Runtime $= O\left(\frac{pT}{p'}\right)$

Page:
Date: / /

* Prefix computation :-
Σ be the domain
operator $\rightarrow \oplus$
eg:- $x, y, z \in \Sigma$
$x \oplus y \in \Sigma$
$y \oplus z \in \Sigma$
$x \oplus z \in \Sigma$

$\rightarrow$ Associativity property :-
$((x \oplus y) \oplus z) = (x \oplus (y \oplus z))$

Let $\Sigma \rightarrow n$ i/p elements
Associative operator $\rightarrow \oplus$
$x_1, x_2, x_3 ---- x_n$
$y_1, y_2, y_3 --- y_n$

(Prefixes of i/p)
$$\begin{cases} y_1 = x_1 \\ y_2 = x_1 \oplus x_2 \\ y_3 = x_1 \oplus x_2 \oplus x_3 = y_2 \oplus x_3 \\ \vdots \\ y_n = x_1 \oplus x_2 \oplus x_3 \oplus \cdots x_n = y_{n-1} \oplus x_n \end{cases}$$

eg:- $\oplus \rightarrow$ addition
I/P :- 3, -5, 8, 2, 5, 4
O/P :- 3, -2, 6, 8, 13, 17

eg:- $\oplus \rightarrow$ multiplication
I/P :- 2, 3, 1, -2, -4
Prefixes :- 2, 6, 6, -12, 48

$\Rightarrow$ For single processor m/c $\rightarrow O(n)$
$\Rightarrow$ For n-processor m/c :- $n=8$, $p=8$ $\oplus \rightarrow$ addition
12, 3, 6, 8, 11, 4, 5, 7
we can use divide and conquer

12, 3	6, 8	11, 4	5, 7
P_1 P_2	P_3 P_4	P_5 P_6	P_7 P_8

Divide the problem into two halves & solve the problem recursively.
Processors need to ~~compute~~ communicate with each other. (Interprocessor comm.)

P_1 P_2
12 3

12, 15	6, 14	11, 15	5, 12

12 15 21 29	11 15 20 27

12, 15, 21, 29, 40, 44, 49, 56

Complexity :- $O(\log n)$.

Step 0 :- If $n=1$, one processor outputs x_1.

Step 1 :- Let the first $n/2$ processors recursively compute the prefixes of $x_1, x_2 \dots x_{n/2}$, and let $y_1, y_2 \dots y_{n/2}$ be the result. At the same time, let the rest of processors recursively compute the prefixes of $x_{n/2+1}, x_{n/2+2} \dots x_n$ and let $y_{n/2+1}, y_{n/2+2} \dots y_n$ be the output.

Step 3 :- The first half of the final answer is the result as $y_1, y_2, \dots y_{n/2}$. The second half of the final answer is $y_{n/2} \oplus y_{n/2+1} \dots y_{n/2} \oplus y_n$.

Page:
Date: / /

$T(1) = 1$ $\quad$ $x_1, x_2, \ldots, x_{n/2}$ | $x_{n/2+1}, \ldots, x_n$

$T(n) = T\left(\frac{n}{2}\right) + O(1)$

$\Rightarrow T(n) = O(\log n)$

$$\boxed{\text{Speedup} = \frac{O(n)}{O(\log n)}}$$

Total work done $= p \cdot O(\log n) = n \cdot O(\log n)$
$= O(p \cdot \log n) = O(n \log n)$

Efficiency $= \dfrac{O(n)}{O(\log n) \cdot p}$

$= \dfrac{O(n)}{O(\log n) \cdot O(n)}$

$= O\left(\dfrac{1}{\log n}\right)$

→ If efficiency $= O(1)$, algorithm is work optimal.

* List Ranking Problem :-

A :- | 5 | 4 | 2 | 0 | 3 | 1 | → gives right node of each values

A[1] A[2] A[3] A[4] A[5] A[6]

A[6] → A[1] → A[5] → A[3] → A[2] → A[4]

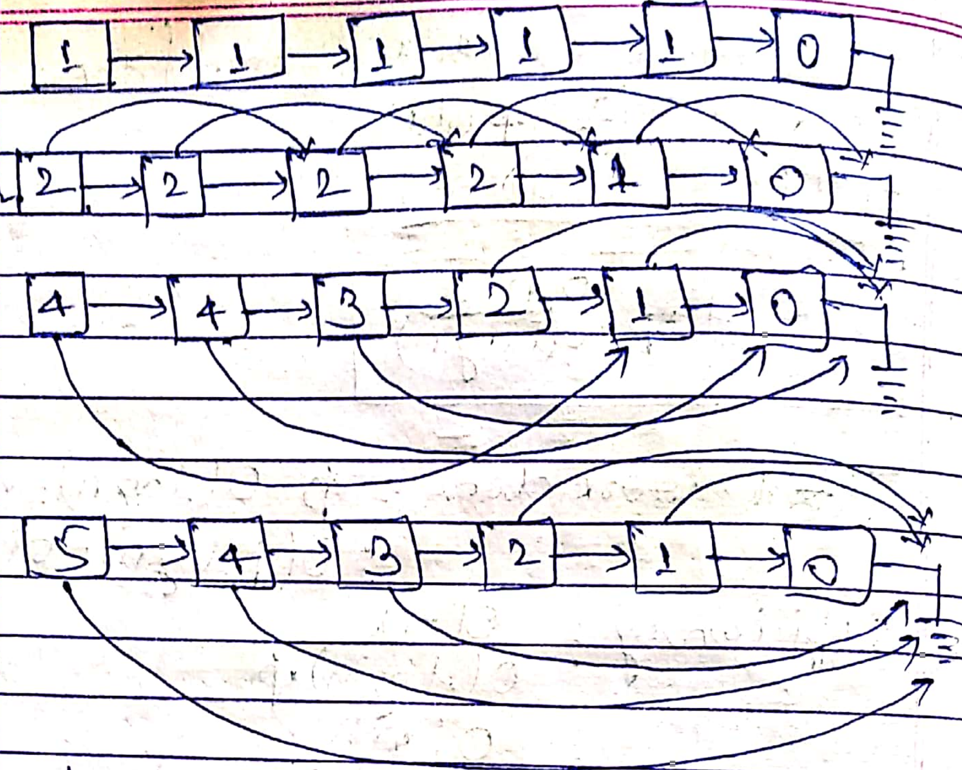
For finding the position from the right side, we use 'pointer jumping' method. i.e., to find the rank of the node.

Initially → Rightmost node have rank 0 and other nodes have rank 1.

Rank[i] = Rank[i] + Rank[Neighbour(i)]
Neighbour[i] = Neighbour[Neighbour[i]]

Date: / /

update Rank & Neighbour



Previously,

A[6] → A[1] → A[5] → A[3] → A[2] → A[4]

Rank :- 5 4 3 2 1 0

Algo :-

for q = 1 to ⌈log n⌉ do
  Processor i (in parallel for $1 \le i \le n$) does
  if (Neighbour[i] ≠ 0)
  { Rank[i] = Rank[i] + Rank[Neighbour[i]]
   Neighbour[i] = Neighbour[Neighbour[i]]
  } → $T(n) = n \log n$

Alt :-

Run these steps until ⌈log n⌉ i.e. ⌈log 6⌉ = 3

	Neighbour	Rank
A :-	5 4 2 0 3 1	1 1 1 0 1 1
	3 0 1 0 2 5	2 1 1 0 2 2
	4 0 0 0 0 2	4 1 2 0 3 4
	0 0 0 0 0 0	4 1 2 0 3 5

But, sequential Runtime, $T(n) = O(n)$.

$\therefore$ speedup $= \dfrac{O(n)}{O(n \log n)} = O\left(\dfrac{1}{\log n}\right)$

Efficiency $= \dfrac{O\left(\frac{1}{\log n}\right)}{n}$

* Problem of selection :-

I/P :- n keys in sequence

find i^{th} smallest element in n keys.

→ Maximum selection with n^2 processors :-

step0 :- If $n=1$, output the key.

step1 :- Processor P_{ij} (for each $1 \le i, j \le n$ in parallel) computes $x_{ij} = (k_i < k_j)$

step2 :- The n^2 processors are grouped into n groups, $G_1, G_2, \ldots G_n$, where G_i $(1 \le i \le n)$ consists of the processors $P_{i1}, P_{i2}, \ldots P_{in}$.

Each group G_i computes Boolean OR of $x_{i1}, x_{i2}, \ldots x_{in}$.

Step3 :- If G_i computes a zero in step2, then Processor P_{i1} outputs k_i as the answer.

$\boxed{T(n) = O(1)}$

eg :- I/P :- $\overset{k_1}{3}, \overset{k_2}{1}, \overset{k_3}{2}, \overset{k_4}{5}, \overset{k_5}{4}$ — $\boxed{k_4 = 5}$ is max.

$n = 5$, $p = 25$

$P_{11} = x_{11} = (k_1 < k_1) \rightarrow 0$	$P_{21} = x_{21} = 1$	$P_{31} = x_{31} = 1$	$P_{41} = x_{41} = 0$	$P_{51} = x_{51} = 0$
$P_{12} = x_{12} = 0$	$P_{22} = x_{22} = 0$	$P_{32} = x_{32} = 0$	$P_{42} = x_{42} = 0$	$P_{52} = x_{52} = 0$
$P_{13} = x_{13} = 0$	$P_{23} = x_{23} = 1$	$P_{33} = x_{33} = 0$	$P_{43} = x_{43} = 0$	$P_{53} = x_{53} = 0$
$P_{14} = x_{14} = 1$	$P_{24} = x_{24} = 1$	$P_{34} = x_{34} = 1$	$P_{44} = x_{44} = 0$	$P_{54} = x_{54} = 1$
$P_{15} = x_{15} = 1$	$P_{25} = x_{25} = 1$	$P_{35} = x_{35} = 1$	$P_{45} = x_{45} = 0$	$P_{55} = x_{55} = 0$
$G_1 = 1$	$G_2 = 1$	$G_3 = 1$	*** $G_4 = 0$ → (k_4 is max)	$G_5 = 1$

→ **Maximum selection with n processors :-**

Step 0 :- If $n = 1$, return k_1.

Step 1 :- Partition the input keys into $\sqrt{n}$ parts $K_1, K_2, K_3 \ldots K_{\sqrt{n}}$, where K_i consists of $k_{(i-1)\sqrt{n}+1}, k_{(i-1)\sqrt{n}+2}, k_{(i\sqrt{n})}$. Similarly, partition the processors so that $P_i (1 \le i \le \sqrt{n})$ consists of the processors $P_{(i-1)\sqrt{n}+1}, P_{(i-1)\sqrt{n}+2}, \ldots P_{i\sqrt{n}}$. Let P_i be the maximum of K_i recursively (for $1 \le i \le \sqrt{n}$)

Step 2 :- If $M_1, M_2, \ldots M_{\sqrt{n}}$ are the group maxima, find and output the maximum of these maxima employing algorithm which computes maximum using n^2 processors. ($\sqrt{n}$ keys & n processors)

eg :- n

$$\underbrace{k_1, k_2, k_3 \ldots k_{\sqrt{n}}}_{K_1}, \underbrace{k_{\sqrt{n}+1} \ldots}_{K_2} \ldots \underbrace{\ldots k_n}_{K_{\sqrt{n}}}$$

$M_1 \quad M_2 \quad M_{\sqrt{n}}$

$P_1 \rightarrow \boxed{P_1, P_2, \ldots P_{\sqrt{n}}}$

$P_2 \rightarrow P_{\sqrt{n}+1}, P_{\sqrt{n}+2} \ldots P_{2\sqrt{n}}$

$n \rightarrow P$
$\sqrt{n} \rightarrow \sqrt{P}$

$$\boxed{T(n) = T(\sqrt{n}) + O(1) = O(\log \log n)}$$

Page :
Date : / /

* Maximum Selection among Integers :-

$[0, n^c] \le c \log n$ (If all integers are in 0 & 1 form)

MSB

$K_1 = 1010$ ×

$K_2 = 1101$ $\Rightarrow$ $\boxed{K_2 = 1101}$

$K_3 = 0110$ × $K_4 = 1100$ ×

$K_4 = 1100$

1st 2nd

* General selection problem :-

$A = K_1 K_2 \ldots K_n$

To find, k^{th} smallest element and

→select(A, K) p processors are given.

Step 1 :- Allocate $\lceil n^x \rceil$ elements to each processor.

$P = n^{1-x}$, where $x < 1$.

Step 2 :- $P_1 = \text{-----}$ M_1

$P_2 = \text{-----}$ M_2

⋮

$P_n = \text{-----}$ M_n

| M_1 | M_2 | ---- | M_n |

↳ median (M)

Step 3 :- Find the median on each processor.

Step 4 :- create an array with M medians.

Step 5 :- Find out the median of the median array.

Step 6 :- L (Nos. less than M) ------ |L|

E (= M) ------ |E|

G (> M) ------ |G|

Step 7 :- (i) if $|L| \ge k$

then select (L, k)

(ii) else if $|L| + |E| \ge k$

then return M

(iii) else select (G, K − (|L| + |E|))

Page :
Date : / /

eg. - A = 8, 9, 33, 15, 22, 11, 16, 25, 37, 35, 28, 3, 7, 29, 21, 35, 33, 32, 25

→ $n = 19$ $P = 4$ $k = 15^{th}$ smallest

$P = n^{1-x}$

$\Rightarrow 4 = 19^{1-x}$

$\Rightarrow x = 1 - \frac{\log 4}{\log 19} = 0.529$

$\lceil n^x \rceil = 19^{0.529} = 4.7 \approx \underline{5}$

		Median		
P_1 → 8, 9, 33, 15, 22	→	15 (M_1)		15 | 25 | 21 | 32
P_2 → 11, 16, 25, 37, 35	→	25 (M_2)		Median → 21 (M)
P_3 → 28, 3, 7, 29, 21	→	21 (M_3)		
P_4 → 35, 33, 32, 25	→	32 (M_4)		

L = 8, 9, 15, 11, 16, 3, 7,

E = 21

→ G = 33, 22, 25, 37, 35, 28, 29, 35, 33, 32, 25

↳ select (G, k − (|L| + |E|)) = $\underline{\text{select}(G, 7)}$

$4 = 11^{1-x}$

$\Rightarrow x = 1 - \frac{\log 4}{\log 11} = 0.421$

$\lceil n^x \rceil = 11^{0.421} = \lceil 2.74 \rceil = 3$

	Median	
P_1 → 33, 22, 25	→ $M_1 = 25$	
P_2 → 37, 35, 28	→ $M_2 = 35$	median 25 (M)
P_3 → 29, 35, 33	→ $M_3 = 33$	
P_4 → 32, 25	→ $M_4 = 25$	

L → 22,

E → 25, 25

G → 33, 37, 35, 28, 29, 35, 33, 32

↳ select (G, 4)

Page :
Date : / /

$2(1-x) = 1$

$4 = 8^{1-x}$

$\Rightarrow 2^2 = 2^{3(1-x)}$

$\Rightarrow 2 = 3 - 3x$

$\Rightarrow 3x = 1 \Rightarrow x = 0.3$

$\lceil 8^{0.3} \rceil = 2$

$P_1 \rightarrow 33, 37 \rightarrow M_1 = 33$
$P_2 \rightarrow 35, 28 \rightarrow M_2 = 28$
$P_3 \rightarrow 29, 35 \rightarrow M_3 = 29$
$P_4 \rightarrow 33, 32 \rightarrow M_4 = 32$

median $M = 29$

$L \rightarrow 28$
$E \rightarrow 29$
$G \rightarrow 33, 37, 35, 35, 33, 32$

↳ select (G, 2)

$4 = 6^{1-x}$

$\Rightarrow x = 1 - \dfrac{\log 4}{\log 6} = 0.226$

$\lceil 6^{0.226} \rceil = \lceil 1.5 \rceil = 2$

$P_1 \rightarrow 33, 37 \rightarrow M_1 = 33$
$P_2 \rightarrow 35, 35 \rightarrow M_2 = 35$
$P_3 \rightarrow 33, 32 \rightarrow M_3 = 32$
$P_4 \rightarrow \times$

Med. $M = 33$

$L \rightarrow 32$
$E \rightarrow 33, 33$
$G \rightarrow 37, 35, 35$

$\because k \leq |L| + |E| \rightarrow$ return M

$\therefore$ 15th smallest $= \underline{33}$

Scanned by CamScanner

Page:
Date: / /

* Merging :-

Given two sorted lists,

$X_1 = k_1, k_2 \ldots k_j \ldots k_m$

$X_2 = k_{m+1}, k_{m+2}, \ldots k_{2m}$

Total RT: → O(log m) (j+q th rank)

Suppose, $2m$ processors are there.

{ m is an integral power of 2
like, $m = 2, 4, 8, \ldots 2^m$ }.

* Odd-even Merge :-

$X_1 = k_1, k_2 \ldots k_m$

$X_2 = k_{m+1}, k_{m+2} \ldots k_{2m}$

$X_1^{odd} = k_1, k_3, \ldots, k_{m-1}$

$X_1^{even} = k_2, k_4, \ldots, k_m$

$X_2^{odd} = k_{m+1}, k_{m+3}, \ldots k_{2m-1}$

$X_2^{even} = k_{m+2}, k_{m+4}, \ldots, k_{2m}$

(recursively) Merge :- X_1^{odd} and X_2^{odd} → $L_1 = l_1, l_2 \ldots l_m$

X_1^{even} and X_2^{even} → $L_2 = l_{m+1}, l_{m+2} \ldots l_{2m}$

Shuffle L_1 and L_2 :-

$l_1, l_{m+1}, l_2, l_{m+2}, l_3, \ldots, l_{2m}$

check (l_{m+i}, l_{i+1}) in order or not.

⇒ $T(m) = T\left(\frac{m}{2}\right) + O(1) = O(\log m)$

eg.:- $X_1 = 2, 5, 8, 11, 13, 16, 21, 25$

$X_2 = 4, 9, 12, 18, 23, 27, 31, 34$

(1) $X_{1,1}$ $X_1^{odd} = 2, 8, 13, 21$

$X_{1,2}$ $X_1^{even} = 5, 11, 16, 25$

$X_{2,1}$ $X_2^{odd} = 4, 12, 23, 31$

$X_{2,2}$ $X_2^{even} = 9, 18, 27, 34$

(2) → $L_1 = 2, 4, 8, 12, 13, 21, 23, 31$

→ $L_2 = 5, 9, 11, 16, 18, 25, 27, 34$

(3)

i) 2, 5, 4, 9, 8, 11, 12, 16, 13, 18, 21, 25, 23, 27, 31, 34

ii) 2, 4, 5, 8, 9, 11, 12, 13, 16, 18, 21, 23, 25, 27, 31, 34

Not include in steps

$y_1 : x_{11}^{odd} = 2, 13$

$x_{11}^{even} = 8, 21$

$y_2 : x_{21}^{odd} = 4, 23$

$x_{21}^{even} = 12, 31$

$y_1^{odd} = 2$

$y_1^{even} = 13$

$y_2^{odd} = 4$

$y_2^{even} = 23$

2, 4

13, 23

2, 4, 13, 23

shuffle

8

21

12

31

8, 12

21, 31

8, 12, 21, 31

2, 8, 4, 12, 13, 21, 23, 31

↳ 2, 4, 8, 12, 13, 21, 23, 31