

Section 1 Q1

Work complexity is the sum of work done by each processor. Time complexity \times no. of processors used

Parallel algo is said to be optimal if it performs the same amount/no. of operations as the best (most optimal) known sequential algorithm.

Section 1 Q4

(i) $X_i =$ ~~no. of people~~ ^{if i^{th} person} picks 1

$$X_i = \begin{cases} 1 & \text{if } i^{\text{th}} \text{ person picks 1} \\ 0 & \text{o/w} \end{cases}$$

$$E[X_i] = P(X_i = 1)$$

$$E[X_i] = \frac{1}{n}$$

$$X = \sum_{i=1}^n X_i = \text{no. of people who pick 1}$$

$$E[X] = E\left[\sum_{i=1}^n X_i\right]$$

$$E[X] = \sum_{i=1}^n E[X_i] \quad (\text{linearity of expectation})$$

$$E[X] = \frac{n}{n} = 1$$

(ii)

~~$Y =$ no. of people who pick~~

$$\rightarrow Y_{ij} = \begin{cases} 1 & \text{if people } i \times j \text{ pick same no.} \\ 0 & \text{o/w} \end{cases}$$

$$E[Y_{ij}] = P(Y_{ij} = 1)$$

$$E[Y_{ij}] = \frac{1}{n}$$

\rightarrow no. of opportunities for 2 ^{specific} people to pick same number =

$$\begin{aligned} & n \text{ people} \\ \Rightarrow & \frac{n(n-1)}{2} \text{ pairs of people} \end{aligned}$$

$$\rightarrow E[\text{no. of events}] = \frac{n(n-1)}{2 \times n} = \frac{n-1}{2}$$

\rightarrow

Section 1 Q3

$$\phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$$

let z_i be indicator variable

| C_i | z_i |
|-------|-------|
| T | 1 |
| F | 0 |

$$C_1 = x_2 \vee \overline{x_3} \vee x_4 = y_2 + 1 - y_3 + y_4$$

$$C_2 = x_1 \vee x_2 \vee \overline{x_4} = y_1 + y_2 + 1 - y_4$$

$$C_3 = x_2 \vee x_4 = y_2 + y_4$$

$$C_4 = x_3 \vee \overline{x_2} = y_3 + 1 - y_2$$

~~x_1, x_2, x_3~~
Subject to C_1

$$\left. \begin{array}{l} \text{C}_1 \quad y_2 + y_4 + 1 - y_3 \geq z_1 \\ \text{C}_2 \quad y_1 + y_2 + 1 - y_4 \geq z_2 \\ \text{C}_3 \quad y_2 + y_4 \geq z_3 \\ \text{C}_4 \quad y_3 + 1 - y_2 \geq z_4 \end{array} \right\} \text{and } y_s \text{ \& } z_s \in \{0, 1\}$$

So, maximise $\sum_i z_i$ \leftarrow OBJECTIVE FUNCTⁿ
~~subject to the following~~
~~Constraints~~
 subject to —

② let C_{i+} be indices of variables in pure form in C_i
 C_{i-} be indices of varⁿ " complemented form in C_i

~~for all i~~
 ② $\sum_{j \in C_{i+}} y_j + \sum_{j \in C_{i-}} (1 - y_j) \geq z_i$ for all i
 where y_j and z_i in $\{0, 1\}$ for all i and j

any 3 Qs

Harshita Sharma
20171099

Section 2 Q1

Accelerated cascading combines a fast but work ~~optimal~~ ^{inefficient} algorithm with a work optimal (slow) algorithm. So, the problem is divided into smaller subproblems, which are first solved (for large size input) by work optimal algo then the sub-results are combined with the faster version of the algorithm.

$$\begin{array}{l|l} A - O(\log n)^T & O(n \log n) \\ B - O\left(\frac{\log n}{\log \log n}\right) & O(n) \end{array}$$

So, $O(\log n)$ time algo

We know that A is non-optimal by a factor of $\log n$. So, we use algo B initially for size n input and when ~~the size of~~ ~~we reduce~~ the size of input ~~it~~ reduces to

$\frac{n}{\log n}$; we use algo A.

i.e. size of inp:

$$\begin{array}{c} n \rightarrow \frac{n}{k} \cdots \rightarrow \frac{n}{\log n} \rightarrow \cdots \\ \underbrace{\hspace{10em}}_{\text{use B}} \quad \underbrace{\hspace{10em}}_{\text{use A}} \\ W_B: O(n) \quad W_A: O(n' \log n') \end{array}$$

where $n' = \frac{n}{\log n}$

$$\Rightarrow W_A: O\left(\frac{n}{\log n} \log\left(\frac{n}{\log n}\right)\right)$$

PTO

$$\Rightarrow W_A : O\left(\frac{n}{\log n} (\log n - \log \log n)\right)$$

$$[\log n - \log \log n < \frac{1}{2} \log n]$$

$$\Rightarrow W_A : O(n)$$

$$\Rightarrow \text{Total Work} = O(n) + O(n) = O(n)$$

$$\Rightarrow \text{Total time} = O\left(\underbrace{\frac{\log\left(\frac{n}{\log n}\right)}{\log \log\left(\frac{n}{\log n}\right)}}_B\right) + \underbrace{O(\log n)}_A$$

~~$O(\log n)$~~

~~$n \log n$~~

Section 2 Q4

(i) ANSV algo — merge 2 sorted arrays of $n/2$ elements each

for (i in 1 to $n/2$) { [do in parallel]

rank [$B[i]$] = ANSV (A , $B[i]$)

rank [$A[i]$] = ANSV (B , $A[i]$)

}

$$W(n) = \cancel{\frac{n}{2}} \frac{n}{2} W\left(\frac{n}{2}\right) + \frac{n}{2} W\left(\frac{n}{2}\right)$$

$$W(n) = n W\left(\frac{n}{2}\right)$$

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

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

(ii) ANSV algo to match parentheses
let S be a seq of parentheses
 $|S| = n$