prob :- 7

The N dimensional fourier eigenbasis be given by:

$$\phi_m = \frac{1}{N}\begin{pmatrix} 1 \\ \exp(2i\pi m/N) \\ \vdots \\ \exp(2i\pi(N-1)m/N) \end{pmatrix}$$

And the set $\mathbb{F}_N = \{\phi_m\}_{m=0}^{N-1}$ is the N point discrete fourier basis.

Here ; the transition from the standard basis to the basis $\mathbb{F}_N$ is called discrete Fourier transform. i.e we go from basis $\mathbb{R}^N$ to the complex basis $\mathbb{F}_N$.

Clearly the N dimensional basis transformation is done by the rule of linear algebra & the change of basis matrix is a $N \times N$ one; that required to do the vector multiplication:

$$X = F_N x \quad \dots (1).$$

Clearly being a $N \times N$ matrix ; the computational complexity is $\cancel{\sim O(n^2) \sim O()}$ $\sim O(N^2)$

In F.F.T ; we break the N dimensional product in eq(1) to four $N/2$ dimensional products like:

$$X = F_N x = \begin{pmatrix} F_{N/2} & D_N F_{N/2} \\ F_{N/2} & -D_N F_{N/2} \end{pmatrix} \begin{pmatrix} x_e \\ x_o \end{pmatrix} \quad \dots (2)$$

When $x_e, x_o \in \mathbb{R}^{N/2}$ are even & odd entries of x; D_N is $\mathbb{R}^{N/2 \times N/2}$ diagonal matrix with diagonal elements $\exp(-2\pi i m/N)$ for $m \in [0, N/2)$

Using the F.F.T; we get the terms of $F_{N/2}x_e$ & $F_{N/2}x_o$ will appear two times in the calculation of $F_N x$.

$$\begin{pmatrix} F_{N/2} & D_N F_{N/2} \\ F_{N/2} & D_N F_{N/2} \end{pmatrix} \begin{pmatrix} x_e \\ x_o \end{pmatrix} = \begin{pmatrix} F_{N/2} x_e + D_N F_{N/2} x_o \\ F_{N/2} x_e - D_N F_{N/2} x_o \end{pmatrix}$$

we have to calculate them for one time & preserve the result for next computation. The only difference between the two elements is the sign before D_N.

Clearly the calculation of $F_{N/2} x_e$ or $F_{N/2} x_o$ needs half the effort (i.e half no of operations) that $F_N x$ use. The ~~mat~~ multiplication of $\frac{N}{2} \times \frac{N}{2}$ 'DIAGONAL' matrix and $\frac{N}{2}$ dimensional vector needs $\frac{N}{2}$ operations (here multiplication). ~~This~~ this gives us the following recurrence relation:

If $T(N)$ represents the no of operations for N point F.F.T then

$$T(N) = \underbrace{2T\left(\frac{N}{2}\right)}_{\text{coming from } F_{N/2} x_e \text{ or } F_{N/2} x_o} + \underbrace{2 \cdot \frac{N}{2}}_{\text{coming from } D_N F_{N/2} x_e \text{ or } D_N F_{N/2} x_o} = 2T\left(\frac{N}{2}\right) + N$$

Now if we do the recursion to reach $T(1)$ from $T(N)$ then we need $\sim \log_2 N$ recursions

as: $2^{\log_2 N} = N$; And at each step, the argument goes to half the previous no.

i.e $\underbrace{T(N) \rightarrow T\left(\frac{N}{2}\right) \rightarrow T\left(\frac{N}{4}\right) \rightarrow \cdots \rightarrow T(1)}_{\log_2 N \text{ times recursion.}}$

But $T(1)$ is $F_1 x^N_{o/e}$ ($x^N_{o/e}$ represents the even or odd entries after $\log_2 N$ recursions) where I've shown that $F_N x$ needs $\sim O(N^2)$ operations.

So $T(1)$; i.e $F_1 x^N_{o/e}$ will need 1^2 i.e 1 operation.

$\therefore$ Finally collecting all ; we get the no of operations for F.F.T algorithm

	recursion		no of operations.
$F_N \longrightarrow$	$T(N)$	$\longrightarrow$	$2T\left(\frac{N}{2}\right) + N$
	$2T\left(\frac{N}{2}\right)$	$\longrightarrow$	$4T\left(\frac{N}{4}\right) + N$
	$4T\left(\frac{N}{4}\right)$	$\longrightarrow$	$8T\left(\frac{N}{8}\right) + N$
	$\vdots$	$\vdots$	$\vdots$
	$2^{\log_2 N} \cdot T\left(\frac{N}{2^{\log_2 N}}\right)$	$\longrightarrow$	$2^{\log_2 N} \times 1$ i.e N

$\log_2 N$ times recursion

(+) $(N + N + \cdots \log_2 N \text{ times})$

$= (N \log_2 N)$

So the complexity of F.F.T is decreased at a level of $\sim O(N \cdot \log_2 N)$

proved