Analysis

→ Correctness

___ $X[i]$ ___

___ $Y[i]$ ___

$X[i] < Y[i]$

already sorted

___ $X[i]$ ___

___ $Y[i]$ ___

$X[i] = Y[i]$

Running time:

If the input consists of n strings, each of length k,
then running time $= \Theta(kn)$

Numbers

If input consists of n strings, each of d digits
in base b, then run time $= \Theta(d(b+n))$

d phaseses:

each phase: count sort of n numbers in $0, \dots, b-1$

Sorting n numbers in $\{1, \dots, n^2\}$ in $O(n)$ time

each number in base n

digits $= 2$

Applying radix sort : $O(2(n+n)) = O(n)$

$$x \in \{1, \dots, n^2\} = (a, b)$$

can be written as

$$x = an + b,$$

$$a \leq n \qquad 0 \leq b \leq n-1$$

Solutions

6. Counting inversions

30 Jan 2025

Polynomial Multiplication

I/P : $f(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$

$g(x) = b_0 + b_1 x + \cdots + b_{n-1} x^{n-1}$

$f : [a_0, a_1, \ldots, a_{n-1}]$

O/P : $f(x) \cdot g(x) = \sum_{i=0}^{2n-2} c_i x^i$

$$c_i = \sum_{j=0}^{i} a_j b_{i-j}$$

To find c_i : $(i+1)$ multiplications $+$ i additions

$$\text{Time} = \sum_{i=0}^{2n-2} (2i+1) = \Theta(n^2)$$

Goal : Improve to $O(n \log n)$

$n = 2$:

$$f = a_0 + a_1 x$$

$$g = b_0 + b_1 x$$

Similar to integer multipli-cation

coeff of x: $(a_0 + a_1) \cdot (b_0 + b_1) - a_0 b_0 - a_1 b_1$

We can find $f \cdot g$ using 3 multiplications.

$$f(0) = a_0 \quad ; \quad g(0) = b_0$$

$$f(1) = a_0 + a_1 \quad ; \quad g(1) = b_0 + b_1$$

coeff of x in $f \cdot g$ $= f(1) \cdot g(1) - f(0) \cdot g(0) - (f(1) - f(0)) \cdot (g(1) - g(0))$

$$f(-1) = a_0 - a_1 \quad ; \quad g(-1) = b_0 - b_1$$

$$f(x) \cdot g(x) = c_0 + c_1 x + c_2 x^2$$

$$c_0 = f(0) \cdot g(0)$$

$$c_1 + c_2 + c_3 = f(1) \cdot g(1)$$

$$c_0 - c_1 + c_2 = f(-1) \cdot g(-1)$$

$\rightarrow$ System of linear equations

$$h(x) = c_0 + c_1 x + \cdots + c_d x^d$$

If we know $h(x_0)$, $h(x_1)$, $\dots$, $h(x_d)$, we can find $c_0, c_1, \dots, c_d$.

$$h(x) = c_0 + c_1 x + c_2 x^2$$

$$h(0) = c_0$$

$$h(1) = c_0 + c_1 + c_2$$

$$h(2) = c_0 + 2c_1 + 4c_2$$

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^d \\ 1 & x_1 & x_1^2 & \cdots & x_1^d \\ & & \vdots & & \\ 1 & x_d & x_d^2 & \cdots & x_d^d \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_d \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_d \end{bmatrix}$$

invertible matrix

$$h(x_i) = y_i$$

<u>Step 1</u>: Find $f(x_1), f(x_2), \ldots, f(x_d)$

$g(x_1), g(x_2), \ldots, g(x_d)$

If $\deg(f) = \deg(g) = n-1$,

Then we need

$$d = 2n - 1$$

to get all coefficients

Step 2 : Find $f(x_1) \cdot g(x_1)$, $f(x_2) \cdot g(x_2)$, ..., $f(x_d) \cdot g(x_d)$

To find $f(x_1) \longrightarrow O(n)$ time ??

$a_0 + a_1 x + a_2 x^2$

$a_0 + x(a_1 + x(a_2))$

Step 1 : $\Theta(n^2)$

Step 2 : $\Theta(n)$

$a_0 + a_1 x + a_2 x^2 + a_3 x^3$

$a_0 + x(a_1 + x(a_2 + x(a_3)))$

Discrete Fourier Transform

$\mathrm{DFT} : \mathbb{C}^n \longrightarrow \mathbb{C}^n$

$\mathrm{DFT}_n(a_0, a_1, \ldots, a_{n-1}) \leftrightarrow a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$

$= f(1), f(\omega_n), \ldots, f(\omega_n^{n-1}) \qquad \omega_n = e^{2\pi i/n}$

$\mathrm{DFT}_3(1, -2, 4)$

$= (3, 1 - 2\omega + 4\omega^2, 1 - 2\omega^2 + \omega)$

$1 - 2x + 4x^2$

$\omega_3 = \omega$

$\omega_{-n} = e^{-2\pi i/n}$

$\mathrm{DFT}_{-n}(a_0, a_1, \ldots, a_n) = (f(1), f(\omega_{-n}), \ldots, f(\omega_{-n}^{n-1}))$

If $DFT_n(a_0, a_1, \dots, a_{n-1}) = (b_0, b_1, \dots, b_{n-1})$

then $\frac{1}{n} DFT_{-n}(b_0, b_1, \dots, b_{n-1}) = (a_0, a_1, \dots, a_{n-1})$

* We can find $DFT_n(a_0, \dots, a_{n-1})$ in $O(n \log n)$ time.

$$f(x) = a_0 + a_1 x + \dots + a_{n-1} x^{n-1}$$

$$= (a_0 + a_2 x^2 + a_4 x^4 + \dots) + x(a_1 + a_3 x^2 + a_5 x^4 + \dots)$$

$$f(x) = f_{even}(x) + x \cdot f_{odd}(x)$$

$$\left.\begin{aligned} f_{even}(y) &= a_0 + a_2 y + a_4 y^2 + \cdots \\ f_{odd}(y) &= a_1 + a_3 y + a_5 y^2 + \cdots \end{aligned}\right\} \text{degrees} \le n/2$$

$$f(1),\ f(\omega_n),\ f(\omega_n^2),\ \ldots,\ f(\omega_n^{n-1})$$

$$f(x) = f_{even}(x^2) + x \cdot f_{odd}(x^2)$$

Recursively find

$$DFT_{n/2}(f_{even}),\quad DFT_{n/2}(f_{odd})$$

$DFT_{n/2}$: $f_{even}(1)$, $f_{even}\left(\frac{\omega_n}{2}\right)$, ... , $f_{even}\left(\frac{\omega_n^n}{2}\right)$
(f_{even})

Need : $f_{even}(1)$, $f_{even}(\omega_n^2)$, $f_{even}(\omega_n^4)$, ...

$$\omega_{\frac{n}{2}} = e^{\frac{2\pi i}{n/2}} = e^{\frac{4\pi i}{n}} = \omega_n^2$$

$$\omega_{n/2}^k = \omega_n^{2k}$$

Given f of degree $n-1$,

① recursively find $DFT_{n/2}(f_{even})$, $DFT_{n/2}(f_{odd})$,
$\frac{n}{2}$ $\frac{n}{2}$

??

② Find $DFT_n(f)$ using n additions

$$T(n) = T(n/2) + O(n)$$

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

Fast Fourier Transform $\longrightarrow$ name of the algorithm to find DFT

$$f(x) = 1 + x + 2x^2 - 4x^3$$

$$g(x) = 2 - 3x + x^2 + x^3$$

deg (prod) = 6

take a (power of 2) ≥ 6

Convenience : work with powers of 2.

$f(1), \; f(\omega_8^1), \; f(\omega_8^2), \; \dots, \; f(\omega_8^7)$

$g(1), \; g(\omega_8^1), \; g(\omega_8^2), \; \dots, \; g(\omega_8^7)$

$$h(x) = f(x) \cdot g(x)$$

Find : $h(1), \; h(\omega_8^1), \; h(\omega_8^2), \; \dots, \; h(\omega_8^7)$

Coefficients of $h(x) = \frac{1}{8} \, DFT_{-8} \left[h(1), h(\omega_8^1), \dots, h(\omega_8^7) \right]$