Lecture 2

Plan for today

Divide and conquer paradigm

High level structure

1. Divide input into smaller instances
2. Solve the smaller instances recursively
3. Combine the solutions somehow to solve the original problem.

Time complexity analysis

- Setup a recurrence relation
- Solve the recurrence ← gen via Master theorem

Master theorem

$T(n): \mathbb{N} \to \mathbb{N}$

s.t. $T(n) \le a \cdot T(n/b) + \Theta(n^d)$

$a, b, \cancel{c}, d$ — constants (do not depend on n)

$a \geq 1$, $b \geq 1$, $d \geq 0$

Ex: Merge Sort

$a = 2$, $b = 2$, $d = 1$, $c = 10$

Today $a = 7$, $b = 2$, $d = 2$

Thm

$$T(n) = \begin{cases} O(n^d \cdot \log n) & \text{if } a = b^d \\ O(n^d) & \text{if } a < b^d \\ O(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

Problem 1: Algorithm for fast integer multiplication

Input: x, y - n-digit natural numbers

Output: 1) $x + y$

2) $x \cdot y$

Naive algorithm:

- Digit by digit
- primary school method.

$$
\begin{array}{r}
56789 \\
+\ 12345 \\
\hline
69134
\end{array}
$$

Running time: $O(n)$

Best that we can hope for.

Best thing we can hope for

Integer Multiplication

Middle school algo.

① Multiply each of the digits of y with x

(2) shift appropriately and add.

$$
\begin{array}{rrrrrrrr}
 & & & 2 & 3 & 3 & & \\
 & & & 5 & 6 & 7 & 8 & \\
 & & & 1 & 2 & 3 & 4 & \\
\hline
 & & 2 & 2 & 7 & 1 & 2 & \\
 & \_ & \_ & \_ & \_ & \_ & 0 & \\
\_ & \_ & \_ & \_ & \_ & 0 & 0 & \\
\_ & \_ & \_ & \_ & \_ & 0 & 0 & 0 \\
\hline
\_ & \_ & \_ & \_ & \_ & & &
\end{array}
$$

Running time: $O(n^2)$

Is this the best that we can hope for?

61 Anatoly Karatsuba ~ $O(n^{1.59\ldots})$

70s Schonhage-Strassen ~ $O(n \log n \cdot \log\log n)$

2019 Harvey - ~~~~ $O(n \log n)$

Karatsuba's Algorithm: Divide and conquer.

① $X = a \cdot 10^{n/2} + b$

$Y = c \cdot 10^{n/2} + d$

$$\begin{array}{r} 5\,6\,7\,8 \\ \times\ 1\,2\,3\,4 \\ \hline \end{array}$$

a, b, c, d – $n/2$ digit integers

② $\underline{X \cdot Y} = \underline{ac} \cdot 10^n + 10^{n/2}(\underline{ad} + \underline{bc}) + \underline{bd}$. — (*)

3) Compute (Recursively)
$a \cdot c$, $b \cdot d$, $a \cdot d$, $b \cdot c$

④ Combine as per (*) to get $X \cdot Y$

✓ $5678 = 999 \times a + b$

$\underline{5678} = \underline{56} \cdot 10^2 + 78$

$1234 = 12 \cdot 10^2 + 34$

- Recursively mult
  $56 \cdot 12$, $56 \cdot 34$
  $78 \cdot 12$, $78 \cdot 34$
- Combine solutions somehow

Running time

$$T(n) \le 4 \cdot T(n/2) + O(n)$$

$$\Rightarrow T(n) \le O(n^{\log_2 4}) = O(n^2)$$

$$X \cdot 10^n = X\underbrace{00\cdots0}_{n}$$

$$X = X_k X_{k-1} \cdots X_1$$

Obs1: $\quad ad + bc = \underline{(a+b)}\,\underline{(c+d)} - \underline{ac} - \underline{bd}$

Pf: Just expand.

Karatsuba's Algorithm

(0) $n=1$ then output $x \cdot y$

(1) $x = a \cdot 10^{n/2} + b \qquad y = c \cdot 10^{n/2} + d$

— $O(n)$

(2) Recursively compute ac, bd, $(a+b)(c+d)$ $\to O(n)$

(3) Use obs 1 to get $ad+bc$ — $O(n)$

(4) Use eqn $(*)$ to get $x \cdot y$. — $O(n)$

Correctness:

Follows from Obs 1 + eqn $(*)$.

Running time:

$$T(n) \leq 3 \cdot T(n/2) + O(n)$$

Master theorem: $T(n) \leq O(n^{\log_2 3})$

Master theorem: $T(n) = O(n^{\log_2 3})$

$$\approx O(n^{1.59\ldots})$$

Formally stating the claim of correctness.

n – power of 2.

Claim: $\forall\, n \in \mathbb{N}$ and all $x, y \in \mathbb{N}$, n-digits

if x, y are inputs to K-algorithm, then the output equals $x \cdot y$.

Pf: By induction on n.

Base case

$n = 1$ — correct due to step 0.

Induction step.

Assume correctness for ~~n/2~~ integers with strictly less than n-digits.

- step 2 is correct due to I.H

- step 2 + obs 1 + Eqn (*)

  $\Rightarrow$ Induction step.

$\boxtimes$

Algorithms for Matrix Multiplication

Input: X, Y — $n \times n$ matrices, integer entries.

Output:

Output $X \cdot Y$

Warm up

$(X+Y)_{i,j} = X_{i,j} + Y_{i,j}$

- Just add entry wise
- $O(n^2)$ operations } can't expect to do better.

$\boxed{\text{Model}}$: - Count the number of integer arithmetic operations
- Ignore the digit information and assume all entries are on a fixed number of digits.

Multiplication

$$X = \begin{array}{c c} & j \\ i & \begin{pmatrix} & x_{i,j} & \end{pmatrix} \end{array} \qquad Y = \begin{array}{c c} & j \\ i & \begin{pmatrix} & y_{i,j} & \end{pmatrix} \end{array}$$

$$(X \cdot Y)_{i,j} = \sum_{k=1}^{n} x_{ik} \cdot y_{kj} \quad \text{— (i)}$$

$$[n] = \{1, 2, \ldots, n\}$$

Algo 1

1) For all $(i,j) \in [n] \times [n]$

- Use equation (i) to compute $(XY)_{i,j}$

integer arithmetic operations

$= O(n^2 \cdot n) = O(n^3)$

can't expect better than $O(n^2)$ running time.

$\longrightarrow$

n^2 $\sim n^{2.3}$ $n^{\log_2 7}$ n^3

Strassen's Algo.   Naive

Algo 2 (Divide and Conquer) $n \sim$ power of 2

$n/2 \times n/2$ blocks

$$X = \begin{pmatrix} A & B \\ & \end{pmatrix}$$

$$\left(\begin{array}{c|c} C & D \end{array}\right)_{n\times n}$$

$$Y = \left(\begin{array}{c|c} E & F \\ \hline G & H \end{array}\right)_{n\times n} \qquad n/2 \times n/2 \text{ blocks.}$$

$$X \cdot Y = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \cdot \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

$$= \left(\begin{array}{c|c} AE+BG & AF+BH \\ \hline CE+DG & CF+DH \end{array}\right) \quad \text{—} (2)$$

Exercise

Verify (2). (follows from the definition of MM)

Natural Recursive Algo.

— Rec. compute AE, BG, CF, ... — ...

— Put them together as per (2)