

lecture 10 CS124

(1)

Divide and Conquer

1. divide
2. conquer
3. combine

usually at low levels
can switch to another alg. to save
overhead

ex: Find min and max of an array

[3, 4, 1, 6, 8]

iterative: $2n$ comparisons

recursive: $T(n) = 2T(\frac{n}{2}) + 2, T(2) = 1$

$T(n) = \frac{3}{2}n - 2$, solve by induction
or Master theorem

ex: integer multiplication

$\Theta(n^2)$

$\begin{array}{r} 12 \mid 34 \\ \times 56 \mid 78 \\ \hline \end{array}$

$x = 10^{n/2}a + b$

$y = 10^{n/2}c + d$

$x \cdot y = 10^n(a \cdot c) + 10^{n/2}(ad + bc) + bd$

$T(n) = 4T(\frac{n}{2}) + O(n)$
← addition is $O(n)$
and multiplication by 10^x
is just bit shifts

Master theorem

$= \Theta(n^2)$ ← doing 4 sub-multiplications

$(a + b)(c + d) = ac + (bc + ad) + bd$
difference

3 multiplications

$T(n) = 3T(\frac{n}{2}) + O(n)$

$\Theta(n^{\log_2 3}) \sim \Theta(n^{1.59})$
 $a = 3, b = 2, k = 1$
 $3 > 2$

recurse down to machine instruction

here algo close to linear $\Theta(n^{1+\epsilon}), \epsilon > 0$

ex. matrix multiplication

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} AE+BG & AF+BH \\ CE+DG & CF+DH \end{bmatrix}$$

assume
n is power of 2

naive $O_{ik} = \sum_j a_{ij} \cdot b_{jk} \quad O(n^3)$

divided

$$T(n) = 8T\left(\frac{n}{2}\right) + O(n^2)$$

$$a=8, b=c, k=2$$

$$8 > 2^2$$

$$O(n^{\log_2 8}) = O(n^3)$$

but we
can get 7 multiplications

Strassen

$$P_1 = A(F-H)$$

$$P_2 = (A+B)H$$

$$P_3 = (C+D)E$$

$$P_4 = D(G-E)$$

$$P_5 = (A+D)(E+H)$$

$$P_6 = (B-A)(G+H)$$

$$P_7 = (A-C)(E+H)$$

$$\begin{bmatrix} P_5 + P_1 & -P_2 + P_6 \\ -P_2 + P_6 & \end{bmatrix}$$

lin. comb

more
operations

$$T(n) = 7T\left(\frac{n}{2}\right) + O(n^2)$$

$$O(n^{\log_2 7}) \approx O(n^{2.807})$$

$$O(n^{2+\epsilon})$$

$$2.796, 2.778, 2.522, \\ 2.517, 2.496, 2.479 \\ 2.376, 2.324, 2.3727$$

Dynamic Programming

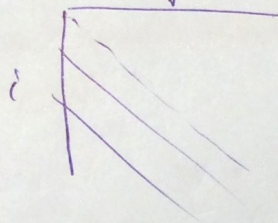
(2)

String Reconstruction

THESE ARE THE REASONS

- 1) ^{greedy}
+ backtracking \rightarrow exp. runtime
- 2) Div. Conc. with fixed ~~splitting~~ ^{splitting} \rightarrow exp. runtime
- 3) ~~exhaustive~~ ^{exhaustive} ~~splitting~~ ^{splitting} ~~splitting~~ \rightarrow looking for subproblem

$O(n^3)$



for $d = 1 \dots n-1$

for $i = 1 \dots n-d$

$j = i+d$

for $k = i+1 \dots j-1$

$D(i,j) = D(i,k) \text{ and } D(k,j)$