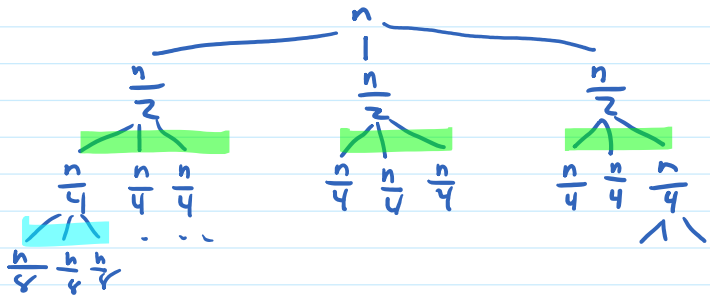
Recursion Tree

Recursion tree for $T(n) = 3T(\frac{n}{2}) + cn$

- 3: # subproblems
- $\frac{n}{2}$: size of subproblems
- cn: divide/combine work



cn $\quad i=0$

$3c \cdot \frac{n}{2}$ $\quad i=1$

$9c \cdot \frac{n}{4}$

$27c \cdot \frac{n}{8}$

$\vdots$

$$\sum_{i=0}^{k} r^i = \frac{r^{k+1}-1}{r-1}$$

$k = \log_2 n + 1 \quad r = \frac{3}{2}$

$$\leq \sum_{i=0}^{\log_2 n + 1} \left(\frac{3}{2}\right)^i \cdot cn$$

$$= cn \sum_{i=0}^{\log_2 n + 1} \left(\frac{3}{2}\right)^i$$

$$\left(\frac{a}{b}\right)^k = \frac{a^k}{b^k}$$

$$= cn \cdot 2 \left(\left(\frac{3}{2}\right)^{\log_2 n + 2} - 1 \right)$$

$$\leq 2cn \cdot \left(\frac{3}{2}\right)^{\log_2 n + 2}$$

$$= \frac{9}{2} cn \left(\frac{3}{2}\right)^{\log_2 n}$$

$$= \frac{9}{2} cn \cdot \frac{3^{\log_2 n}}{2^{\log_2 n}}$$

$$= \frac{9}{2} cn \cdot \frac{3^{\log_2 n}}{n}$$

$$= \frac{9}{2} c \cdot 3^{\frac{\log_3 n}{\log_3 2}}$$

$$= \frac{9}{2} c \cdot \left(3^{\log_3 n}\right)^{\frac{1}{\log_3 2}}$$

$$= \frac{9}{2} c \, n^{\frac{1}{\log_3 2}}$$

$$= \frac{9}{2} c \cdot n^{\log_2 3}$$

$= \frac{1}{2} c \cdot n^{\log_2 3}$

$\in O(n^{\log_2 3})$ $\log_2 3 < 2$

Master Method

Suppose $T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$

(a: # subproblems; $\frac{n}{b}$: size of subproblems; $f(n)$: work done for divide + combine)

Then n^d $d < \log_b a$

if $f(n) \in O(n^{\log_b a - \varepsilon})$ for some $\varepsilon > 0$ then $T(n)$ is $\Theta(n^{\log_b a})$

$d = \log_b a$

if $f(n) \in \Theta(n^{\log_b a})$ then $T(n)$ is $\Theta(f(n) \cdot \log n)$

if $f(n) \in \Omega(n^{\log_b a + \varepsilon})$ for some $\varepsilon > 0$ and if $a \cdot f\left(\frac{n}{b}\right) \le c \cdot f(n)$ for some $c < 1$ and all large n

$T(n)$ is $\Theta(f(n))$

Examples: $T(n) = \underset{a}{9}\, T\left(\frac{n}{\underset{b}{3}}\right) + n$

$\log_b a = \log_3 9 = 2$ $\quad n$ is $O(n^{2 - 0.01})$ $\quad$ so $T(n)$ is $\Theta(n^{\log_b a}) = \Theta(n^2)$

$T(n) = \underset{a=1}{} T\left(\underset{b=\frac{3}{2}}{\frac{3}{2}} n\right) + 1$

$\log_b a = \log_{\frac{3}{2}} 1 = 0$ $\quad 1$ is $\Theta(n^0)$ $\quad$ so $T(n)$ is $\Theta(\log n)$

$T(n) = \underset{a}{3} \cdot T\left(\frac{n}{\underset{b}{4}}\right) + n \log n$

$\log_b a = \log_4 3$ $\quad n \log n$ is $\Omega(n^{\log_4 3 + 0.01})$ $\quad$ so $T(n)$ is $\Theta(n \log n)$

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

$\log_b a = \log_2 2 = 1$ $\quad n \log n$ is not $\Theta(n)$

is not $O(n^{1-\varepsilon})$ for any ε

is not $\Omega(n^{1+\varepsilon})$ for any ε

M.M (as given) does not apply

($\exists$ more powerful versions)

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

n is $\Theta(n^{\log_2 2})$ $\quad T(n)$ is $\Theta(n \log n)$

Binary search on array $T(n) = T(\frac{n}{2}) + c$

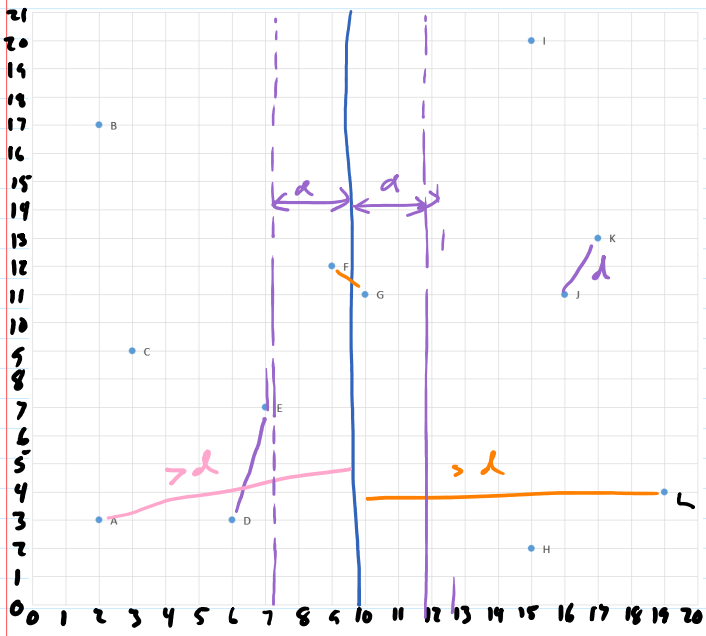
c is $\Theta(n^{\log_2 1})$ $T(n)$ is $\Theta(\log n)$

Binary search on linked list (silly) $T(n) = T(\frac{n}{2}) + cn$

n is $\Omega(n^{\log_2 1})$ $T(n)$ is $\Theta(n)$

Karatsuba integer mult $T(n) = 3T(\frac{n}{2}) + cn$

n is $O(n^{\log_2 3 - 0.01})$ so $T(n)$ is $\Theta(n^{\log_2 3})$



Closest Pair: given n pts in 2-D plane
find 2 that are closest together

brute force: compute dist between all pairs
$\Theta(n^2)$

0) preprocess: sort by x, sort by y — $O(n \log n)$

Divide and conquer

1) divide pts into leftmost $\frac{n}{2}$, rightmost $\frac{n}{2}$ } sorted by both x (easy!) and y (think of how to avoid resorting)

2) find closest in each half

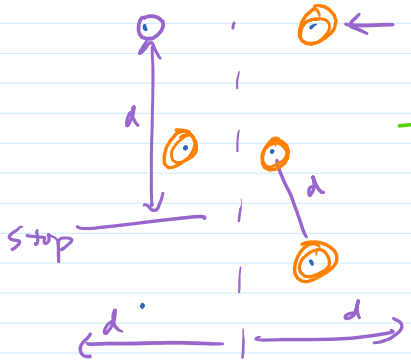
3) find closest of those 2 closest pairs
↳ let dist be d

4) find pts within d horizontally of midline

5) for each p in middle strip — $O(n)$ iterations
for each q below p in order of y-coord and until vert. distance > d — $\Theta(1)$
compute dist(p, q)
update closest over all if < d

6) return closest pair ever seen

$T(n) = 2 \cdot T(\frac{n}{2}) + c(n)$
$T(n)$ is $O(n \log n)$



max 7 other pts with d of mid line and d vertically from curr pt in mid