Lecture 24

This week:

- TRACE evals
- Remote Lectures
- No Recitations
- Remote OH

Tue 6PM
Thu 7PM

final project WED 12/8

Hw PB4 Sat 12/11

- MergeSort vs QuickSort ✓
- Order of growth, asymptotic notation Θ, O, Ω
- Solve Recurrences for asymptotic growth
  - later in Algo. course

MergeSort-Rec $T(n)$ RunTime

2 8 7 1 3 5 6 4

$\boxed{2\ 8\ 7\ 1}$ $\boxed{3\ 5\ 6\ 4}$

- split
- Rec Calls MergeSort-Rec(Left) $T(n/2)$: 1 2 7 8
  MergeSort-Rec(Right) $T(n/2)$ 3 4 5 6
- Merge (non-Rec) $\Theta(n)$ linear
  already sorted halves

L, R

$\boxed{1\ 2\ 3\ 4\ 5\ 6\ 7\ 8}$

$\Downarrow$

output

$$\boxed{T(n) = \Theta(n) + 2T\left(\frac{n}{2}\right)}$$

Run Time Recurrence.

RT QuickSort-Rec

2 8 7 1 3 5 6 4

2	1	3	$\boxed{4}$	7	5	6	8

Left: val ≤ pivot — p: final position — Right: val > pivot

- $\theta(n)$ • non-Rec pivoting/partition
  Left/Right are now completely separated
- $T(p)$ • QuickSort-Rec(Left) size = p
- $T(n-p-1)$ • QuickSort-Rec(Right) size n-p-1

order depends on partition procedure (exercise)

R.T. Rec $T(n) = T(p) + T(n-p-1) + n$

Left(Rec) Right(Rec) partition

~~random p=Unknown~~

Avg/Expectation with uniform-dis over p

$p = 1, 2, \ldots, n-1$ prob uniform $1/n$

$$\boxed{E[T(n)] = \frac{1}{n} \sum_{p=1}^{n-1} \left[n + T(p) + T(n-p-1) \right]}$$

Asymptotic Growth

$$f(n) = \Theta(g(n))$$

if f bounded by $g \cdot$ const

$$\overbrace{c_1 g(n) \le f(n)}^{\text{lower bound for } f} \underbrace{\le c_2\, g(n)}_{\text{upper bound for } f}$$

constants > 0

examples

$f(n) = 2n^2 + n - 1 = \Theta(n^2)$ $\quad g(n) = n^2$

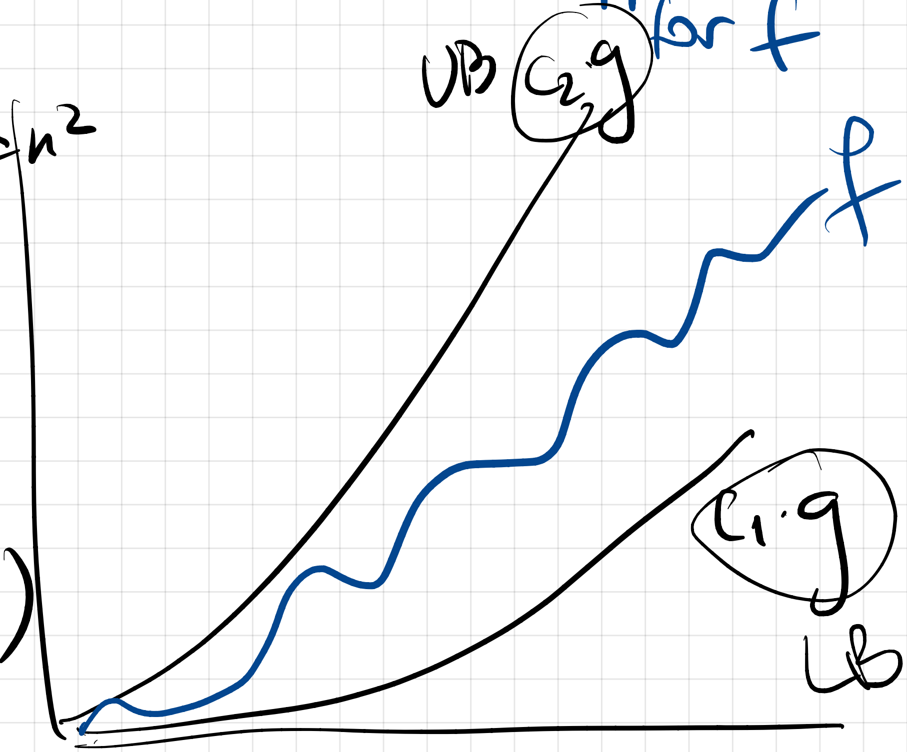
$$\underbrace{1}_{c_1} n^2 \le 2n^2 + n - 1 \le \underbrace{3}_{c_2} \cdot n^2$$

$$f(n) = n - 6\log n + 2 = \Theta(n)$$

$$\underbrace{\tfrac{1}{2}}_{c_1} n \le n - 6\log n + 2 \le \underbrace{1}_{c_2} n$$

$$f(n) = 2^n + n^2 + 3 = \Theta(2^n)$$

$1 \cdot 2^n \le$ $\qquad \le 2^n \cdot \overset{c_2}{2}$



Θ = tight asympt bound (both UB, LB)

$$\underbrace{c_1 \cdot g(n) \le f(n)}_{\text{only lower bound}} \le c_2\, g(n)$$

$$f(n) \underbrace{\le c_2\, g(n)}_{\text{only upper bound}}$$

both bounds $\Theta(g(n))$

only lower bound

"big omega" $f(n) = \Omega(g(n))$

"at least $g(n)$"

LB

only upper bound

$f(n) \le c_2 \cdot g(n)$

"big O" $f(n) = O(g(n))$

"at most $g(n)$"

UB

$f = \Theta(g) \iff \begin{cases} f = \Omega(g) & \text{LB} \\ \boxed{\text{and}} & \\ f = O(g) & \text{UB} \end{cases}$

tight bound

Solving Recurrences (Simple)

Rec 1) Binary Search $T(n)$ $\quad T(n) = 1 + T(n/2)$

- check the middle $\Theta(1)$ const
- recurse on one side $T(n/2)$

iterations / grinding

$k=1$ $T(n) = 1 + T(n/2)$ apply rec for $n/2$

$k=2$ $= 1 + [1 + T(n/4)] = 2 + T(n/4)$

apply rec for $n/4$

$k=3$ $= 2 + [1 + T(n/8)] = 3 + T(n/8)$

$k=4$ $= 3 + [1 + T(n/16)] = 4 + T(n/16)$

in general $\dots$

k iter $\quad k + T(n/2^k)$

↑ general (k) iteration pattern ↑

. . . .

when does it stop? args inside $T \simeq 1$

$\boxed{\text{Last } k:}$ $T\left(n/2^k\right) \simeq T(1) \Leftrightarrow \frac{n}{2^k} \simeq 1 \Leftrightarrow k \simeq \log(n)$

$\log(n)$

$$T(n) = \log(n) + T\left(\frac{n}{2^{\log(n)}}\right) = \log(n) + T(1)$$

const

$$= \log(n) + \underline{\text{const}} = \Theta(\log n)$$

Rec 2 Selection Sort

Repeat $T(n)$
- find min $\Theta(n)$
- consider rest of array $T(n-1)$

$T(n) = n + T(n-1)$

iterations

$k=1$ $\quad T(n) = n + T(n-1) =$

$k=2$ $\quad = n + [(n-1) + T(n-2)] = n + (n-1) + T(n-2)$

$k=3$ $\quad = n + (n-1) + [(n-2) + T(n-3)] = n + (n-1) + (n-2) + T(n-3)$

. . . .

general k

$= n + (n-1) + (n-2) \dots + (n-k+1) + T(n-k)$

. . . .

Last k? How does it end? want $T(n-k) = T(0)$

$n-k \simeq 0 \iff \boxed{k \simeq n}$

$$T(n) = \underbrace{n + (n-1) + (n-2) \dots + (n-n+1)}_{\text{reverse}} + T(0)$$

$$1+2+3 \dots + n + T(0)$$

const

$$= \frac{n(n+1)}{2} + T(0) = \frac{n^2+n}{2} + T(0) = \Theta(n^2)$$

growth bounds: $\frac{1}{2} n^2 \leq \frac{n^2+n}{2} \leq 1 n^2$

Rec3 Merge Sort

$T(n)$ $\qquad$ $T(n) = n + 2T(n/2)$

- MergeSort(Left) $\quad T(n/2)$ size $= n/2$
- MergeSort (Right) $\quad T(n/2)$ size $= n/2$
- Combine/Interleave/Merging two sides $\quad \Theta(n)$

$k=1 \quad T(n) = n + 2\underline{T(n/2)}$

$k=2 \quad = n + 2\left[\frac{n}{2} + 2T\left(\frac{n}{4}\right)\right] = 2n + 4T\left(\frac{n}{4}\right)$

$k=3 \quad = 2n + 4\left[\frac{n}{4} + 2T\left(\frac{n}{8}\right)\right] = 3n + 8T\left(\frac{n}{8}\right)$

$k=4 \quad = 3n + 8\left[n/8 + 2T\left(\frac{n}{16}\right)\right] = 4n + 16T\left(\frac{n}{16}\right)$

general after k iterations

$$= kn + 2^{k} T\left(\frac{n}{2^k}\right)$$

Last k

How it ends?

want $T\left(\frac{n}{2^k}\right) \simeq T(1) \Longleftrightarrow k \simeq \log n$

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

$$= \underbrace{n \cdot \log(n)}_{\Theta(n \log n)} + \underbrace{n \cdot T(1)}_{\Theta(n)}$$

Con

bigger

$$= \boxed{\Theta(n \cdot \log n)}$$

Theorem

OPT sorting time in general case

In practice most popular: Quicksort

QS worst case (pivot at extreme all the time) $\Theta(n^2)$

QS AVG case (pivot at random over many runs) $\Theta(n \log n)$

OPTIMA serial case

QS > Mergesort for practical reasons (memory management)

Rec 4 $T(n) = n + 4T(n/2)$ $k=1$

$k=2$ $= n + 4\left[n/2 + 4T(n/4) \right] = n + 2n + 16T(n/4)$

$k=3$ $= n + 2n + 16\left[n/4 + 4T(n/8) \right] = n + 2n + 4n + 64\boxed{T(n/8)}$

$k=4$ $n + 2n + 4n + 64\left[\frac{n}{8} + 4T\left(\frac{n}{16}\right) \right] = n + 2n + 4n + 8n + 256T\left(\frac{n}{16}\right)$

general pattern after k iter

$n + 2n + 4n \dots + 2^{k-1}n + 4^k T\left(\frac{n}{2^k}\right)$

$= n(1 + 2 + 4 \dots + 2^{k-1}) + 4^k T\left(\frac{n}{2^k}\right)$

geom series $= 2^k - 1$

$= n(2^k - 1) + 4^k T(n/2^k)$

Last k $\frac{n}{2^k} \simeq 1 \iff k = \log n$

$$T(n) = n \cdot \left(2^{\log(n)} - 1\right) + 4^{\log(n)} T\left(\frac{n}{2^{\log n}}\right)$$

$$= n\,(n-1) + \left(2^{\log(n)}\right)^2 \cdot T\left(\frac{n}{n}\right)$$

$$= n^2 - n + n^2 \cdot T(1)$$ const

$$= \Theta(n^2)$$

Solve recurrences

① guess the asymptote.

run few iterations? pattern too difficult, so make a guess.

② prove by induction

$$T(n) = n + 4T\left(\frac{n}{2}\right) \qquad \boxed{\text{guess}} \quad T(n) = \Theta(n^2)$$

want: $\underbrace{\text{const}\cdot n^2 \le T(n)}_{\text{lower bound}} \overbrace{\le \text{const}\cdot n^2}^{\text{upper bound}}$

Ind step for Lower Bound: $T(n/2) \ge c(n/2)^2 \Rightarrow T(n) \ge cn^2$

Proof: $T(n) = n + 4T\left(\frac{n}{2}\right) \overset{\text{Ind Hyp}}{\ge} n + 4\cdot c\left(\frac{n}{2}\right)^2 =$

$$= n + 4\cdot\frac{cn^2}{4} = cn^2 + n \ge cn^2 \checkmark$$

Ind Step for Upper Bound $\boxed{\cancel{T\left(\frac{n}{2}\right) \le c\left(\frac{n}{2}\right)^2 \Rightarrow T(n) \le cn^2}}$ does not work

need stronger statement $\Leftarrow$ (ind proof)

TOO LOOSE, CAN'T make the proof

Ind Step UB $T(n) \le cn^2 - dn$ subtract lower term (sufficient)

$$\boxed{T(n/2) \underset{\text{ind hyp}}{\le} c\left(\frac{n}{2}\right)^2 - d\left(\frac{n}{2}\right) \Rightarrow T(n) \le cn^2 - dn}$$ works!

Proof $T(n) = n + 4T\left(\frac{n}{2}\right) \overset{\text{Ind hyp}}{\le} n + 4\left(c\left(\frac{n}{2}\right)^2 - d\left(\frac{n}{2}\right)\right) =$

$= n + \cancel{cn^2} - 2dn \overset{\text{want}}{\le} \cancel{cn^2} - dn$

$n \overset{\text{want}}{\le} +2dn - dn \quad \Big| \div n$

$1 \overset{\text{want}}{\le} d$ ✓ set constant $d > 1$

Master Th (Simple Version) for Recurrences.

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$$

a calls Rec

$\frac{n}{b}$ size of input each call

non-Rec part

3 cases:

- $c < \log_b a \implies T(n) = \Theta(n^{\log_b a})$
- $c = \log_b a \implies T(n) = \Theta(n^{\log_b a} \cdot \log(n))$
- $c > \log_b a \implies T(n) = \Theta(n^c)$

MT: $T(n) = aT(\frac{n}{b}) + \Theta(n^c)$

MS $\quad T(n) = 2T(n/2) + n \quad a=2, b=2, c=1$

BS $\quad T(n) = T(n/2) + 1 \quad a=1, b=2, c=0$