Lecture 2 Divide and Conquer 1

Monday, January 13, 2020 9:59 AM

- analyze running time of divide-and-conquer.

merge-sort(a[])

0 base case

1 split a[] into b[], c[]

2 merge-sort(b[]), merge-sort(c[]) recursive call

3 merge(b[], c[])

merge cost

cost of steps 0 1 3

can be analyzed directly

recursion cost

Cost of step 2

- to analyze the running time

let $T(n)$ be the running time of the alg.

for input of size n.

- analyze merge cost : function of n
- analyze recursion cost : written as $T(k)$ for some $k < n$

- $T(n)$ = merge cost + recursion cost.

- analyzing merge-sort

- merge cost $O(n)$
- recursion cost $\underbrace{2}_{\#\text{recursive calls}} \times T(\underbrace{\tfrac{n}{2}}_{\text{problem size for each call}})$

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

base case $T(1) = 0$

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

1 guess-and-verify

guess: $T(n) \le Cn\log_2 n$

verify: induction

hypothesis: $T(n) \le \boxed{Cn\log_2 n}$

verify: induction

hypothesis: $T(n) \le \boxed{Cn\log_2 n}$

base case: $n=1$ $\quad T(1) = 0 = C\,1\cdot\log_2 1$

true for every C

induction: suppose IH is true for all $k < n$

will show IH is also true for n

$$T(n) = 2T\left(\tfrac{n}{2}\right) + n \qquad \text{(uses recursion)}$$

$$\le 2\cdot\left(C\cdot\frac{n}{2}\log_2\frac{n}{2}\right) + n \qquad (IH)$$

$$= \cancel{2}\, C\cdot\frac{n}{\cancel{2}}\cdot(\log_2 n - 1) + n$$

$$= \boxed{Cn\log_2 n} - C\cdot n + n$$

want $T(n) \le Cn\log_2 n$

need $Cn\log_2 n - Cn + n \le Cn\log_2 n$

$$C \ge 1$$

whenever $C \ge 1$, we have $T(n) \le Cn\log_2 n$ $\square$

$T(n) \le n\log_2 n$, $T(n) = O(n\log n)$

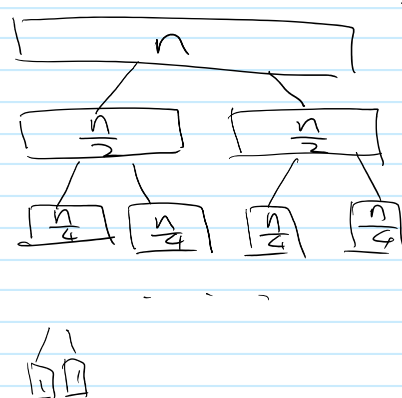
(2) recursion tree

draw a tree of all recursive calls

node $\longleftrightarrow$ recursive call

edge $\longleftrightarrow$ one calls the other

leaf $\longleftrightarrow$ base case



#nodes	size	level
1	n	0
2	$\frac{n}{2}$	1
4	$\frac{n}{4}$	2
2^i	$\frac{n}{2^i}$	i
n	1	$\log_2 n$

Claim: $T(n) =$ sum of merge cost for every node on the recursion tree

for merge-sort

$$T(n) = \sum_{i=0}^{\log_2 n - 1} \text{merge cost for level } i$$

$$= \sum_{i=0}^{\log_2 n - 1} 2^i \times \frac{n}{2^i}$$

2^i: # nodes at level i

$\frac{n}{2^i}$: merge cost for each node at level i

$\log_2 n - 1$

$$= \sum_{i=0} n = \underline{n \log_2 n}$$

- interpret recursion tree method

$$T(n) = 2\underline{T\left(\tfrac{n}{2}\right)} + n$$

$$= 4T\left(\tfrac{n}{4}\right) + 2 \cdot \tfrac{n}{2} + n \qquad \boxed{T\left(\tfrac{n}{2}\right) = 2 \cdot T\left(\tfrac{n}{4}\right) + \tfrac{n}{2}}$$

$$= 8T\left(\tfrac{n}{8}\right) + 4 \cdot \tfrac{n}{4} + 2 \cdot \tfrac{n}{2} + n$$

merge cost for layer 2   merge cost for layer 1   merge cost for layer 0

. . .

$$= \sum_{i=0}^{\#\text{layers}} \text{merge cost for layer } i$$