Lec 8/29

Tuesday, August 29, 2017 12:56

to ensure each step takes at least one step

$$\sum_{i=1}^{3n^5} \left(\frac{2n^3}{i} + 1 \right) \in \Theta(n^5)$$

even if body of for loop doesn't run, checking & incrementing takes time.

Divide-and-conquer algorithms ?

Euclid's algorithm

runtime proportional to # recursive calls.

the remainders reduce by at least $\frac{1}{2}$ every 2 steps.

so runtime $\in \Theta(\log n)$ where $n = \min(a, b)$

Show $c = a \bmod b \leq \frac{1}{2} a$. This is true since if $b \leq \frac{1}{2} a$ then $a \bmod b < b \leq \frac{1}{2} a$

if $b > \frac{1}{2} a$ then $a = \underset{n \geq 1}{n} b + c \Rightarrow c = a - nb < a - \frac{1}{2} a = \frac{1}{2} a$

assuming $a_0 \geq b_0$, since $T(n) \in O(\log(a))$,

if $b \neq 0$. $T(n) \in O(\log(b))$ too since b becomes a after 1 step.

Now Divide and conquer:

```
function DAC(x):
    if x small enough
        solve directly
    else
        divide into smaller cases x_1, ..., x_n
        y_i <- DAC(x_i)
        y <- combine(y_1, ..., y_n)
        return y
```

$$T(|x|) = \cdots + \sum_{i=1}^{n} T(|x_i|) + \cdots$$

When x_i's are same size, $x_1, \ldots, x_k$ are about of size $\lfloor \frac{n}{b} \rfloor$ when $|x| = n$,

$$T(n) = \begin{cases} c & \text{if } n \leq n_0 \\ k\,T(\lfloor \frac{n}{b} \rfloor) + f(n) & \text{else} \end{cases}$$

Array version of mergesort

Procedure MergeSort $(A[i, \dots, j])$

```
if i = j :
    return
m ← ⌊(i+j)/2⌋
Mergesort (A[i, m])
mergesort (A[m+1, j])
merge (A[i, m], A[m+1, j])
```

merging takes liner time.

$$T(n) = T(\lfloor \tfrac{n}{2} \rfloor) + T(\lceil \tfrac{n}{2} \rceil) + \Theta(n)$$

When n a power of 2, say 2^b

$$\begin{aligned} T(n) &= 2T(\tfrac{n}{2}) + \Theta(n) \\ &= 2^b T(1) + \sum_{i=1}^{b} \Theta(n) \\ &= \Theta(n) + \Theta(n \log n) \\ &= \Theta(n \log n) \end{aligned}$$

Solving recurrences

- Iteration
- recurrence tree
- guess and prove??