Test #1

we'll start
lecture @ 7:15

50min

- Next week (Feb 9/10), **in your registered tutorial timeslot**
- Room assignments will be sent out on Monday
- Coverage: Weeks 1 - 4 (Induction, Recurrences)

"cheat sheet" 1 single sided 8.5" x 11"
handwritten.

Outline

Wrap up RecBinSearch

MergeSort

Divide and Conquer

Notes

Recursive Binary Search

Upper bound on $T(n)$

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 1 + \max(T(\lceil n/2 \rceil), T(\lfloor n/2 \rfloor)) & n > 1 \end{cases}$$

We were showing $T(n) \in \Theta(\lg n)$.

We showed $T(n) \in \Omega(\lg n)$.

Need: $T(n) \in O(\lg n)$ ←

$T(n)$ is non-decr. ←

Inductive Step: Show $T(n) \le \underline{\lg(n-1)} + 2$ (from our experimental last time)

Let $n > 2$

Assume $H(n)$: $\forall i \in \mathbb{N}, 2 \le i < n, T(i) \le \lg(i-1) + 2$

Show $H(n) \rightarrow C(n)$: $T(n) \le \lg(n-1) + 2$

$$T(n) = 1 + \max(T(\lceil n/2 \rceil), T(\lfloor n/2 \rfloor)) \quad \text{(by def)}$$

$$= 1 + T(\lceil n/2 \rceil) \quad \text{(b/c } T \text{ is non-decreasing)}$$

$$\le 1 + \lg(\lceil n/2 \rceil - 1) + 2 \quad \text{(by } H(n), \ 2 \le \lceil n/2 \rceil < n \text{ for } n > 2)$$

$$\le 1 + \lg\left(\frac{n-1}{2}\right) + 2 \quad \left(\text{by } \lceil n/2 \rceil - 1 \le \frac{n+1}{2} - 1 = \frac{n-1}{2}\right)$$

$$\le \cancel{1} + (\lg(n-1) - \cancel{\lg} 2) + 2 = \lg(n-1) + 2.$$

Base Case: $T(2) = 1 + T(1) = 2 \le \lg(1) + 2 = 2$ (with $\lg(1) = 0$)

Conclude: $T(n) \in O(\lg n)$

$\Rightarrow T(n) \in \Theta(\lg n)$

Recursive Binary Search

$T(n)$ is non-decreasing $\rightarrow$ Prove by Complete Induction!

$P(n)$: $\forall m \in \mathbb{N}$, $m < n$ then $T(m) \leq T(n)$

Inductive Step: Let $n \geq 1$

Assume $H(n)$: $\forall i \in \mathbb{N}$, $1 \leq i < n$, $P(i)$

Show $H(n) \rightarrow C(n)$: $P(n)$

Case: $n = 1 \rightarrow$ trivial, no $m < 1$

Case: $n = 2 \rightarrow T(n) = 1 + \max(T(\lceil n/2 \rceil), \cancel{T(\lfloor n/2 \rfloor)}) = 2$

$T(2) \geq T(1)$ ✓

Case: $n > 2$ By $H(n)$, $P(\lceil n/2 \rceil)$, $P(\lfloor n/2 \rfloor)$, $P(n-1)$ hold.

$1 \leq \lfloor \frac{n-1}{2} \rfloor < n$

Sufficient to show that $T(n-1) \leq T(n)$

$$T(n-1) = 1 + \max\left(T\left(\left\lceil \tfrac{n-1}{2} \right\rceil\right) \overset{\geq}{,} T\left(\left\lfloor \tfrac{n-1}{2} \right\rfloor\right)\right) \leq 1 + T\left(\left\lceil \tfrac{n-1}{2} \right\rceil\right)$$

(by $H(n)$)

$$\underset{\text{by } H(n)}{\leq} 1 + T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) \underset{\text{by } H(n)}{=} 1 + \max(T(\lceil n/2 \rceil), T(\lfloor n/2 \rfloor)) = T(n)$$ ✓

Recurrence for MergeSort

$n = e - b + 1$

sort

```
MergeSort(A,b,e):
    if b == e:  return
    m = (b + e) / 2
```

1 1

```
    MergeSort(A,b,m)
    MergeSort(A,m+1,e)
```

$\lceil n/2 \rceil$ $\lfloor n/2 \rfloor$

merge

```
    # merge sorted A[b..m] and A[m+1..e] back into A[b..e]
    for i in [b,...,e]:  B[i] = A[i]
```

n

```
    c = b
    d = m+1
```

n

```
    for in [b,...,e]:
        if d > e or (c <= m and B[c] < B[d]):
            A[i] = B[c]
            c = c + 1
        else:  # d <= e and (c > m or B[c] >= B[d])
            A[i] = B[d]
            d = d + 1
```

Recurrence for MergeSort

Worst case running time of MergeSort satisfies

$$T(n) = \begin{cases} 1 \text{ (first line)} & \text{if } n=1 \\ 1 \text{ (constant stuff)} \\ + T(\lceil n/2 \rceil) \text{ (1st recursive call} \\ + T(\lfloor n/2 \rfloor) \text{ (2nd recursive call)} & n > 1 \\ + n \text{ (1st loop to copy)} \\ + n \text{ (2nd loop to merge)} \end{cases}$$

$$T(n) = \begin{cases} 1 & \text{if } n=1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \underbrace{2n+1} & \text{if } n>1 \end{cases} \quad \theta(n)$$

↳ this could be some other linear function

Unwind $T(n)$ Simplifying assumption $n = 2^k$

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \\ 2\cdot T(n/2) + 2n + 1 & \text{if } n > 1 \end{cases}$$

$$T(n) = T(2^k) = 2\cdot \underbrace{T(2^{k-1})} + 2\cdot 2^k + 1$$

$$= 2\cdot\left[2\cdot T(2^{k-2}) + 2\cdot 2^{k-1} + 1\right] + 2\cdot 2^k + 1$$

$$= 4\cdot T(2^{k-2}) + \cancel{2^{k+1} + 2^{k+1}}\; 2\cdot 2\cdot 2^k + 3$$

$= \ldots$ after i steps

$$T(2^k) = 2^i \cdot T(2^{k-i}) + 2\cdot i\cdot 2^k + (2^i - 1)$$

Base case reached when $2^{k-i} = 1$ $\quad i = k = \lg n$

$$T(2^k) = \underbrace{2^k \cdot T(1)}_{n} + 2\cdot k\cdot 2^k + 2^k - 1 \qquad T(1) = 1$$

$$= n + 2\cdot n\cdot \lg n + n - 1$$

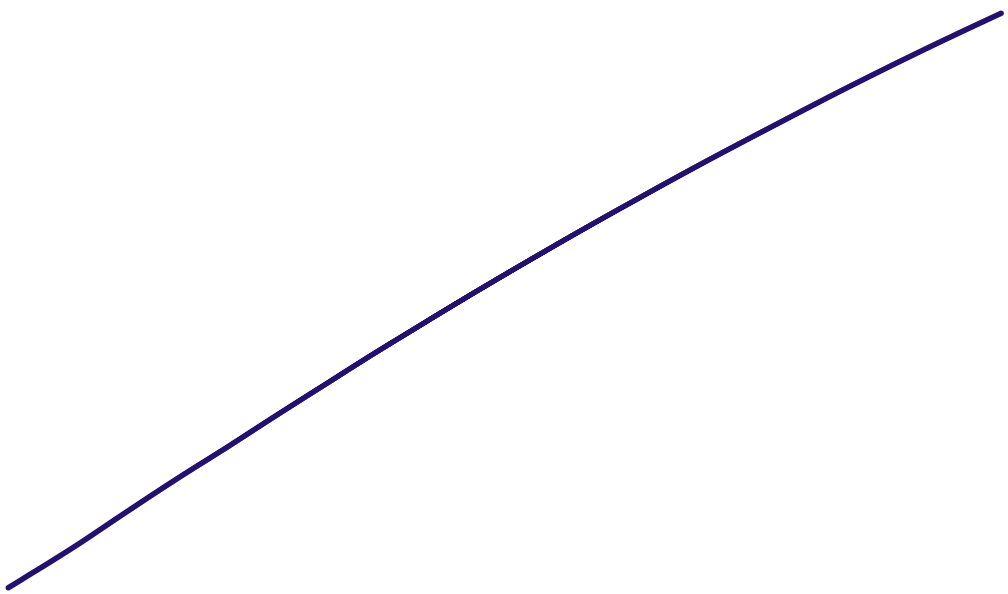
$$T(n) = 2\cdot n\cdot \lg n + 2n - 1$$

Prove $T(n) \in \Theta(n \lg n)$

Show $T(n) \in \Omega(n \lg n)$

and $T(n) \in O(n \lg n)$

Unwind $T(n)$



Merge Sort

Lower bound on $T(n)$

Prove $T(n) \in \Omega(n \lg n)$

* sketch: pieces will be missing for now

Try $T(n) \geq \underbrace{2 \cdot n \cdot \lg n}$ $\forall n \geq 1$ by Complete Ind.

want $\lfloor n/2 \rfloor, \lceil n/2 \rceil \geq 1$
so set $n \geq 2$ ↳ so covered by IH

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 2n + 1$$

$$\geq 2 \cdot \lceil \tfrac{n}{2} \rceil \cdot \lg(\lceil n/2 \rceil) + 2 \cdot \lfloor n/2 \rfloor \cdot \lg(\lfloor n/2 \rfloor) + 2n + 1$$

$$\geq 2 \cdot (\lceil \tfrac{n}{2} \rceil \cdot \underline{\lg(\lceil n/2 \rceil)} + \lfloor n/2 \rfloor \cdot \underline{\lg(\lfloor n/2 \rfloor)}) + 2n + 1$$

$$\geq 2 \cdot \lg(\lfloor \tfrac{n}{2} \rfloor) \cdot (\lceil \tfrac{n}{2} \rceil + \lfloor \tfrac{n}{2} \rfloor) + 2n + 1 \quad \text{(bc lg is increasing)}$$

$$\geq 2 \cdot \lg(\lfloor n/2 \rfloor) \cdot n + \underline{2n} + 1$$

$$\geq 2n(\lg(\lfloor \tfrac{n}{2} \rfloor) + 1) + 1$$

$$\geq 2n(\lg(\lfloor n/2 \rfloor + \lg 2) + 1 \quad \text{(since } \lg 2 = 1\text{)}$$

$$\geq 2n(\lg(2 \cdot \lfloor \tfrac{n}{2} \rfloor)) + 1 \quad \text{(lg identity)}$$

Merge Sort

Lower bound on $T(n)$

$$\hookrightarrow \geq 2n \cdot \lg\left(2 \cdot \frac{n-1}{2}\right) + 1$$

$$2 \cdot \lfloor \tfrac{n}{2} \rfloor \geq 2 \cdot \frac{n-1}{2}$$

$$\geq 2n \cdot \lg(n-1) + 1$$

$\geq n$

Want to get rid of -1.

$$\lceil n/2 \rceil + 1 \geq \lfloor n/2 \rfloor + 1 \geq \frac{n-1}{2} + 1 = \frac{n+1}{2}$$

So we'll try $T(n) \geq 2 \cdot n \cdot \lg(n+1)$

Let's try base case first $T(1) = 1$

$$2 \cdot 1 \cdot \lg(1+1) = 2$$

$1 \not\geq 2$

So we'll try $T(n) \geq 2 \cdot n \cdot \lg(n+1) - 1$

Will writeup Ω proof to post.

Merge Sort

Lower bound on $T(n)$

(added after class)

Proof by Complete Induction that $T(n) \geq 2 \cdot n \cdot \lg(n+1) - 1, \ \forall n \in \mathbb{N}, n \geq 1$

Inductive Step: Let $n \in \mathbb{N}, n > 1$

Assume $H(n)$: $\forall i \in \mathbb{N}, 1 \leq i < n, T(i) \geq 2i \lg(i+1) - 1$

Show $H(n) \rightarrow C(n)$: $T(n) \geq 2n \lg(n+1) - 1$

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 2n + 1 \qquad \text{(by def.)}$$

$$\geq 2 \cdot \lceil n/2 \rceil \cdot \lg(\lceil n/2 \rceil + 1) \cancel{-1} + 2 \cdot \lfloor n/2 \rfloor \cdot \lg(\lfloor n/2 \rfloor + 1) - 1 + 2n + \cancel{1} \qquad \left(\text{by } H(n), \text{ since } \lfloor n/2 \rfloor, \lceil n/2 \rceil \geq 1\right)$$

$$\geq 2 \cdot \lg\left(\frac{n+1}{2}\right) \cdot (\lceil n/2 \rceil + \lfloor n/2 \rfloor) + 2n - 1 \qquad \left(\lceil n/2 \rceil + 1 \geq \lfloor n/2 \rfloor + 1 \geq \frac{n-1}{2} + 1 \geq \frac{n+1}{2}\right)$$

$$= 2 \cdot \lg\left(\frac{n+1}{2}\right) \cdot n + 2n - 1 \qquad (\lceil n/2 \rceil + \lfloor n/2 \rfloor = n)$$

$$= 2 \cdot n \cdot (\lg(n+1) - \lg 2) + 2n - 1 \qquad \text{(lg identity)}$$

$$= 2n \cdot (\lg(n+1) - \cancel{\lg 2 + 1}) - 1$$

$$= 2n \lg(n+1) - 1$$

Base Case: $T(1) = 1 \geq 1 = 2 \cdot 1 \cdot \lg(1+1) - 1$

Conclude: $T(n) \geq 2n \lg(n+1) - 1 \ \ \forall n \in \mathbb{N}, n \geq 1$

Hence, $T(n) \in \Omega(n \lg n)$

Merge Sort

Upper bound on $T(n)$

Merge Sort

Upper bound on $T(n)$

Divide and Conquer: General Case

Class of algorithms: partition problem into b *roughly* equal subproblems, solve, and recombine:

$$T(n) = \begin{cases} k & \text{if } n \leq B \\ a_1\, T(\lceil n/b \rceil) + a_2\, T(\lfloor n/b \rfloor) + f(n) & \text{if } n > B \end{cases}$$

where $B, k > 0$, $b > 1$, $a_1, a_2 \geq 0$, and $a = a_1 + a_2 > 0$. $f(n)$ is the cost of splitting and recombining.

Master Theorem

If f from the previous slide has $f \in \theta(n^d)$, then

$$T(n) \in \begin{cases} \theta(n^d) & \text{if } a < b^d \\ \theta(n^d \log n) & \text{if } a = b^d \\ \theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

Applying the Master Theorem

MergeSort

Applying the Master Theorem

RecBinSearch

Notes