I. Core / B. Divide and Conquer

- General introduction (aspired to on Monday but actually Wednesday)
- Solving recurrences (also **today**)
- The master method (next week Monday)
- Quick sort (next week Wednesday)

Today:

- Old bits from Section 3.1
- Problem 3-4.c
- Common functions (Section 3.2)
- Divide and conquer big pictures (Sections 4.(1 & 2))
- The substitution method (Section 4.3)

For next time

Read sections 4.(4 & 5).

Do Ex 4.5-1 and Problem 4-1.(a.b)

Problem 3-4.c. If $f(n) = O(g(n))$ and $\lg(g(n)) \geq 1$ and $f(n) \geq 1$ for sufficiently large n, then $\lg(f(n)) = O(\lg(g(n))$.

Scratch work: We need a d such that

$$
\begin{aligned}
\lg c + \lg g(n) &\leq d \lg g(n) \\
d &\geq \frac{\lg c}{\lg g(n)} + \frac{\lg g(n)}{\lg g(n)} \\
&\geq \lg c + 1 \\
(\lg c + 1) \lg g(n) &= \lg c \cdot \lg g(n) + \lg g(n)
\end{aligned}
$$

Proof. *Suppose $f(n) = O(g(n))$. Then there exist c, n_0 such that for all $n > n_0$, $f(n) \leq c \cdot g(n)$. Then*

$$
\begin{aligned}
lg\, f(n) &\leq lg\, c\, g(n) && \textit{since lg is increasing} \\
&\leq lg\, c + lg\, g(n) && \textit{by log property} \\
&\leq lg\, c \cdot lg\, g(n) + lg\, g(n) && \textit{Since } lg\, g(n) \geq 1 \\
&\leq (lg\, c + 1) \cdot lg\, g(n)
\end{aligned}
$$

Thus for $n > n_0$, $\lg(f(n)) \leq (lg\, c + 1) \lg(g(n))$.

Big “morals” of §4.(1 & 2)

- Many problems have good divide and conquer solutions. The running time of a divide and conquer algorithm can be captured by a recurrence. So, let’s make sure we can do recurrences.
- Sometimes it’s divide-and-conquer even when it doesn’t seem like it is.
- “Solving” a recurrence means finding an equivalent non-recursive formula.

“Normal” math induction:

“Normal” math induction:

$$\begin{array}{rl} & I(0) \\ & I(n) \rightarrow I(n+1) \\ \therefore & \forall\ n \in \mathbb{N}, I(n) \end{array}$$

“Strong” math induction:

$$\begin{array}{rl} & I(0) \\ & (\forall\ i \leq n, I(i)) \rightarrow I(n+1) \\ \therefore & \forall\ n \in \mathbb{N}, I(n) \end{array}$$

Elements of recurrences (things to look for in making a good guess):

- The coefficient of the recursive application (number of subproblems)
- The divisor of n in the recursive application (size of subproblems)
- The non-recursive terms

Ex. 4.3-1. $T(n) = T(n-1) + n$.

Ex. 4.3-1. $T(n) = T(n-1) + n$. Guess $T(n) \leq c \cdot n^2$. Then

$$\begin{aligned} T(n) &\leq c(n-1)^2 + n \\ &= cn^2 - 2cn + c + n \\ &= cn^2 + (1-2c)n + c \\ &\leq cn^2 \end{aligned}$$

The last step holds as long as

$$\begin{aligned} (1-2c)n + c &\leq 0 \\ (2c-1)n &\geq c \\ n &\geq \tfrac{c}{2c-1} \end{aligned}$$

The recurrence holds so long as $c > \frac{1}{2}$ and $n_0 > \frac{c}{2c-1}$.

4.3-2. $T(n) = T(\lceil \frac{n}{2} \rceil) + 1.$

4.3-2. $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$. First attempt. Guess $T(n) \le c \lg n$

$$\begin{aligned} T(n) &\le c \lg \lceil \tfrac{n}{2} \rceil + 1 \\ &\le c \lg (\tfrac{n}{2} + \tfrac{1}{2}) + 1 \\ &= c \lg (\tfrac{n+1}{2}) + 1 \\ &= c(\lg(n+1) - \lg 2) + 1 \\ &= c(\lg(n+1) - 1) + 1 \\ &= c \lg(n+1) - c + 1 \end{aligned}$$

We would need this to be less than $c \lg n$ ...

4.3-2. $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$. Try again. This time, guess $T(n) \leq c \lg(n - b)$.

$$\begin{aligned} T(n) &\leq c \lg(\lceil \tfrac{n}{2} \rceil - b) + 1 \\ &\leq c \lg(\tfrac{n}{2} + \tfrac{1}{2} - b) + 1 \\ &= c \lg(\tfrac{n+1-2b}{2}) + 1 \\ &= c(\lg(n + 1 - 2b) - \lg 2) + 1 \\ &= c \lg(n + 1 - 2b) - c + 1 \\ &\leq c \lg(n - b) \end{aligned}$$

The last part holds if $n + 1 - 2b \leq n - b$, so $b \geq 1$; and if $-c + 1 \leq 0$, so $c \geq 1$.

4.3-6. $T(n) = 2T(\lfloor \frac{n}{2} \rfloor + 17) + n.$

4.3-6. $T(n) = 2T(\lfloor \frac{n}{2} \rfloor + 17) + n$. Guess $cn \lg n$. Then

$$\begin{aligned}
T(n) &= 2T(\lfloor \tfrac{n}{2} \rfloor + 17) + n \\
&\leq 2c(\lfloor \tfrac{n}{2} \rfloor + 17) \lg(\lfloor \tfrac{n}{2} \rfloor + 17) + n \\
&\leq 2c(\tfrac{n}{2} + 17) \lg(\tfrac{n}{2} + 17) + n \\
&= c(n + 34)(\lg(n + 34) - 1) + n \\
&= cn \lg(n + 34) - cn - c34 + n
\end{aligned}$$

This isn't working out.

4.3-6. $T(n) = 2T(\lfloor \frac{n}{2} \rfloor + 17) + n$. Try again, this time guess $c(n-34)\lg(n-34)$.

$$
\begin{aligned}
T(n) &= 2T(\lfloor \tfrac{n}{2} \rfloor + 17) + n \\
&\leq 2c(\lfloor \tfrac{n}{2} \rfloor + 17 - 34)\lg(\lfloor \tfrac{n}{2} \rfloor + 17 - 34) + n \\
&\leq 2c(\tfrac{n}{2} + 17 - 34)\lg(\tfrac{n}{2} + 17 - 34) + n \\
&= c(n-34)\lg(\tfrac{n-34}{2}) + n = c(n-34)(\lg(n-34) - 1) + n \\
&= c(n-34)\lg(n-34) - cn + 34c + n \leq c(n-34)\lg(n-34)
\end{aligned}
$$

The last step holds if $-cn + 34c + n \leq 0$.

$$
\begin{aligned}
cn - 34c &\leq n \\
c &\geq \tfrac{n}{n-34}
\end{aligned}
$$

Notice that as n gets bigger, the ratio gets closer to 1, but will always be slightly bigger. Pick $c = 2$. Then we need $2n - 68 \geq n$, or $n \geq 68$.

4.3-9. $T(n) = 3T(\sqrt{n}) + \lg n$.

4.3-9. $T(n) = 3T(\sqrt{n}) + \lg n$. Let $m = \lg n$, $n = 2^m$. Then define

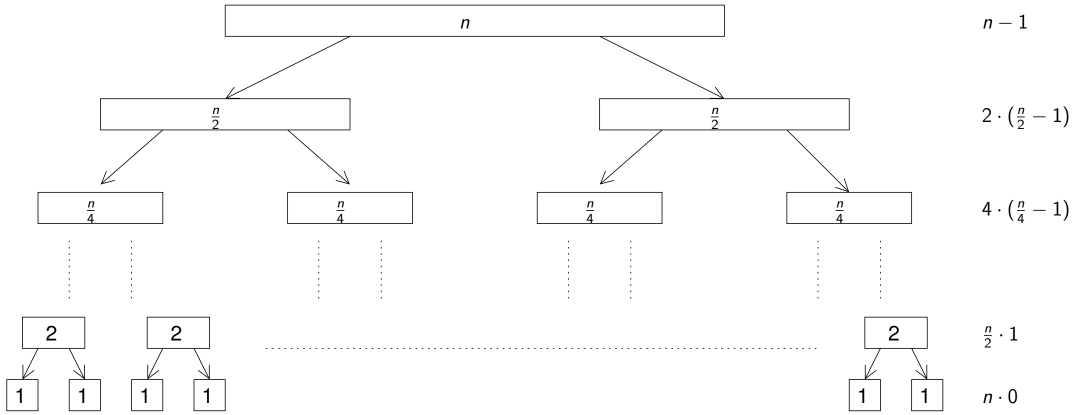
$$\begin{aligned} S(m) &= T(2^m) \\ &= 3T(2^{\frac{m}{2}}) + \lg 2^m \\ &= 3T(2^{\frac{m}{2}}) + m \\ &= 3S(\tfrac{m}{2}) + m \end{aligned}$$

What do you do with that? Guess $cm \lg m$, on the intuition of its similarity to mergesort.

$$\begin{aligned} &= 3c\tfrac{m}{2}\lg\tfrac{m}{2} + m \\ &= \tfrac{3}{2}cm\lg m - \tfrac{3}{2}cm + m \end{aligned}$$

This isn't working out. In fact, the complexity class is wrong.

$$C_{ms}(n) = \begin{cases} 0 & \text{if } n \leq 1 \\ n - 1 + 2C_{ms}(\frac{n}{2}) & \text{otherwise} \end{cases}$$



4.3-9. $T(n) = 3T(\sqrt{n}) + \lg n$. Again, let $m = \lg n$, $n = 2^m$, and $S(m) = 3S(\frac{m}{2}) + m$. Then guess $m^{\lg 3} - \frac{m}{2}$. (Of course.)

$$\begin{aligned}
S(m) &= 3S(\tfrac{m}{2}) + m \\
&= 3((\tfrac{m}{2})^{\lg 3} - \tfrac{m}{2}) + m \\
&= 3\frac{m^{\lg 3}}{2^{\lg 3}} - \frac{3}{2}m + m \\
&= 3\frac{m^{\lg 3}}{3} + \frac{-3+2}{2}m \\
&= m^{\lg 3} - \frac{m}{2}
\end{aligned}$$

So, $S(m) = \Theta(m^{\lg 3}) = \Theta((\lg n)^{\lg 3})$.