Key fact for solving recurrences is understanding geometric series.

Examples:

$$1 + \frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \cdots + \frac{1}{2^n} = O(1)$$

$$1 + 3 + 3^2 + 3^3 + \cdots + 3^n = O(3^n)$$

For constant $\alpha > 0$,

$$\sum_{i=0}^{k} \alpha^i = 1 + \alpha + \alpha^2 + \cdots + \alpha^k$$

if $\alpha < 1$ the 1st term dominates

if $\alpha > 1$ the last term dominates

Lemma: For $\alpha > 0$,

$$\sum_{i=0}^{k} \alpha^i = \begin{cases} O(1) & \text{if } \alpha < 1 \\ O(k) & \text{if } \alpha = 1 \\ O(\alpha^k) & \text{if } \alpha > 1 \end{cases}$$

In the case $a > 1$ we'll have something like:

$3^{\log_2 n}$ for $a = 3$, $k = \log_2 n$

We want to simplify this into a polynomial n^c for constant c.

$\log n = \log_2 n$ when base is not specified it's base 2

$\ln n = \log_e n$

$\log_{10} n = O(\log n)$ & $\ln n = O(\log n)$

So $O(n \log n)$ ← the base doesn't matter

but in the exponent it does:

$2^{\log n} = n$ but is $2^{\log_3 n} = O(n)$? No!

Reexpressing $2^{\log_3 n}$:

Recall $(x^y)^z = x^{yz} = (x^z)^y$

For $2^{\log_3 n}$ ← want to get these 2 numbers to match

Note: $2 = 3^{\log_3 2}$

Thus, $2^{\log_3 n} = \left(3^{\log_3 2}\right)^{\log_3 n} = \left(3^{\log_3 n}\right)^{\log_3 2} = n^{\log_3 2}$

now they match

$\log_3 2 \approx 1.585$

Let's solve a few recurrences we've seen so far:

Slow multiply algorithm: $T(n) = 4T(\frac{n}{2}) + O(n)$

& $T(1) = O(1)$

Thus for some constant $c > 0$,

$T(n) \leq 4T(\frac{n}{2}) + cn$ & $T(1) \leq c$

notice that: $T(\frac{n}{2}) \leq 4T(\frac{n}{2^2}) + c\frac{n}{2}$

Plugging that in we get:

$$T(n) \leq 4T(\tfrac{n}{2}) + cn$$
$$\leq 4^2 T(\tfrac{n}{2^2}) + 4cn/2 + cn$$
$$= 4^2 T(\tfrac{n}{2^2}) + cn\left(1 + \tfrac{4}{2}\right)$$

plugging in $T(\frac{n}{2^2}) \leq 4T(\frac{n}{2^3}) + cn/2^2$

$$T(n) \leq 4^3 T(\tfrac{n}{2^3}) + cn\left(1 + \tfrac{4}{2} + \left(\tfrac{4}{2}\right)^2\right)$$

repeating this for i rounds we get:

$$T(n) \leq 4^i T(\tfrac{n}{2^i}) + cn\left(1 + \tfrac{4}{2} + \left(\tfrac{4}{2}\right)^2 + \cdots + \left(\tfrac{4}{2}\right)^{i-1}\right)$$

stop at $i = \log_2 n$ then $\frac{n}{2^i} = 1$ & $T(\frac{n}{2^i}) = T(1) \leq c$

$$T(n) \leq 4^{\log_2 n} c + cn\left(1 + \tfrac{4}{2} + \left(\tfrac{4}{2}\right)^2 + \cdots + \left(\tfrac{4}{2}\right)^{\log_2 n - 1}\right)$$

$$= O(4^{\log n}) + O(n) \times O(2^{\log n})$$
$$= O(n^2) + O(n) \times O(n) = O(n^2)$$

MergeSort: $T(n) = 2T(\frac{n}{2}) + O(n)$

Thus, $T(n) \leq 2T(\frac{n}{2}) + cn$, $T(1) \leq c$

$$T(n) \leq cn + 2\left(2T(\tfrac{n}{2^2}) + cn/2\right)$$

$$= 2^2 T(\tfrac{n}{2^2}) + cn(1+1)$$

$$\leq 2^2\left(2T(\tfrac{n}{2^3}) + cn/2^2\right) + cn(1+1)$$

$$= 2^3 T(\tfrac{n}{2^3}) + cn(1+1+1)$$

$$\leq 2^i T(\tfrac{n}{2^i}) + icn$$

stop when $i = \log_2 n$

$$\leq 2^{\log_2 n} c + cn\log_2 n$$

$$= O(n) + O(n\log n)$$

$$= O(n\log n)$$

In general, for constants $a>0$, $b>1$,
the recurrence: $T(n)=aT\left(\frac{n}{b}\right)+\Theta(n)$

$$T(n)\le aT\left(\frac{n}{b}\right)+cn$$

$$\le a\left(aT\left(\frac{n}{b^2}\right)+c\left(\frac{n}{b}\right)\right)+cn$$

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

$$\le a^2\left(aT\left(\frac{n}{b^3}\right)+c\left(\frac{n}{b^2}\right)\right)+cn\left(1+\frac{a}{b}\right)$$

$$=a^3T\left(\frac{n}{b^3}\right)+cn\left(1+\left(\frac{a}{b}\right)+\left(\frac{a}{b}\right)^2\right)$$

$$\le a^iT\left(\frac{n}{b^i}\right)+cn\left(1+\left(\frac{a}{b}\right)+\left(\frac{a}{b}\right)^2+\cdots+\left(\frac{a}{b}\right)^{i-1}\right)$$

stop when $i=\log_b n$

$$\le a^{\log_b n}c\cancel{\Theta}+cn\left(1+\left(\frac{a}{b}\right)+\left(\frac{a}{b}\right)^2+\cdots+\left(\frac{a}{b}\right)^{\log_b n-1}\right)$$

$$=O\left(a^{\log_b n}\right)+O(n)\times O\left(\left(\frac{a}{b}\right)^{\log_b n}\right)$$

$$=O\left(a^{\log_b n}\right)$$

$$a^{\log_b n}=\left(b^{\log_b a}\right)^{\log_b n}=\left(b^{\log_b n}\right)^{\log_b a}=n^{\log_b a}$$

$$=O\left(n^{\log_b a}\right)$$

$$T(n) = 2T\left(\frac{n}{3}\right) + O(1)$$

$$T(n) \leq 2T\left(\frac{n}{3}\right) + c, \quad T(1) \leq c$$

$$T(n) \leq 2\left(2T\left(\frac{n}{3^2}\right) + c\right) + c$$

$$= 2^2 T\left(\frac{n}{3^2}\right) + c(1+2)$$

$$\leq 2^2\left(2T\left(\frac{n}{3^3}\right) + c\right) + c(1+2)$$

$$= 2^3 T\left(\frac{n}{3^3}\right) + c\left(1+2+2^2\right)$$

$$\leq 2^i T\left(\frac{n}{3^i}\right) + c\left(1+2+2^2+\cdots+2^{i-1}\right)$$

for $i = \log_3 n$

$$\leq 2^{\log_3 n} c + c\left(1+2+2^2+\cdots+2^{\log_3 n - 1}\right)$$

$$= O\left(2^{\log_3 n}\right) + O\left(2^{\log_3 n}\right) = O\left(2^{\log_3 n}\right)$$

$$= O\left(n^{\log_3 2}\right)$$ (See Page 2)

The Master Theorem:

For constants $a > 0$, $b > 1$, $d \geq 0$,

the recurrence:

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

Solves to:

$$T(n) = \begin{cases} O(n^d) & \text{if } \frac{a}{b^d} < 1 \quad (\text{i.e., } d > \log_b a) \\ O(n^d \log n) & \text{if } \frac{a}{b^d} = 1 \quad (d = \log_b a) \\ O(n^{\log_b a}) & \text{if } \frac{a}{b^d} > 1 \quad (d < \log_b a) \end{cases}$$

Proof idea:

Expanding out as before we get:

$$T(n) \leq cn^d \left(1 + \frac{a}{b^d} + \left(\frac{a}{b^d}\right)^2 + \cdots + \left(\frac{a}{b^d}\right)^{\log n}\right)$$

if $\frac{a}{b^d} < 1$ then 1st term dominates so $T(n) = O(n^d)$

if $\frac{a}{b^d} = 1$ then every term $= 1$, so $T(n) = O(n^d \log n)$

if $\frac{a}{b^d} > 1$ then last term dominates so

$$T(n) = O\left(n^d \left(\frac{a}{b^d}\right)^{\log_b n}\right)$$

$$= O\left(a^{\log_b n}\right)$$

$$= O\left(n^{\log_b a}\right)$$

Example recurrences:

- Binary search: $T(n) = T\left(\frac{n}{2}\right) + O(1)$

$a=1, b=2, d=0$ so $\frac{a}{b^d} = 1$

$T(n) = O(\log n)$

- Merge Sort: $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

$a=2, b=2, d=1$ so $\frac{a}{b^d} = \frac{2}{2} = 1$

So: $T(n) = O(n \log n)$

- Multiplication slow: $T(n) = 4T\left(\frac{n}{2}\right) + O(n)$

$a=4, b=2, d=1$

$\frac{a}{b^d} = \frac{4}{2} > 1$

$T(n) = O(n^{\log_2 4}) = O(n^2)$

- Multiplication fast: $T(n) = 3T\left(\frac{n}{2}\right) + O(n)$

$a=3, b=2, d=1$

$\frac{a}{b^d} = \frac{3}{2} > 1$

$T(n) = O(n^{\log_2 3})$

Some other useful recurrences:

- $T(n) = 2T(\frac{n}{2}) + O(1)$

$a=2,\ b=2,\ d=0$ so $\frac{a}{b^d} = \frac{2}{1} > 1$

$$T(n) = O(n^{\log_2 2}) = O(n)$$

- $T(n) = T(\frac{3}{4}n) + O(n)$

$a=1,\ b=\frac{4}{3},\ d=1$

$$\frac{a}{b^d} = \frac{3}{4} < 1$$

$$T(n) = O(n)$$

Note: for any $b > 1$,

$$T(n) = T(\frac{n}{b}) + O(n)$$

$a=1,\ b \geq 1,\ d=1$

$$\frac{a}{b^d} = \frac{1}{b} < 1$$

$$T(n) = O(n)$$

$O()$ notation:

for functions $f(n)$ & $g(n)$,

if there is a constant $c>0$ so that:

$$f(n) \leq cg(n)$$

then we write $f(n) = O(g(n))$

Example: $f(n) = 10n + 500\sqrt{n} + 2000$

$g(n) = n^2$

then for $c = 3000$, $f(n) \leq cg(n)$

So $f(n) = O(n^2)$

also for $h(n) = n$, for $c = 3000$ $f(n) \leq ch(n)$

So $f(n) = O(n)$

$\Omega()$ notation:

if there is a constant $c>0$ so that:

$$f(n) \geq cg(n)$$

then $f(n) = \Omega(g(n))$

For example: $f(n) = 10n + 500\sqrt{n} + 2000$

$f(n) = \Omega(n)$ & $f(n) = \Omega(\sqrt{n})$

if $f(n) = O(g(n))$ & $f(n) = \Omega(g(n))$

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

Example: $f(n) = 10n + 500\sqrt{n} + 2000$

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