1 Asymptotic Notation

Definitions, abbreviated [1] for this class

$f = \Theta(g)$

f grows at the same rate as g

$\exists\ n_0$ and $c_1, c_2 > 0$ s.t.

$\forall\ n > n_0, c_1 g(n) \leq |f(n)| \leq c_2 g(n)$

$f = O(g)$

f grows no faster than g

$\exists\ n_0$ and $c > 0$ s.t.

$\forall\ n > n_0, |f(n)| \leq cg(n)$

$f = \Omega(g)$

f grows at least as fast as g

$\exists\ n_0$ and $c > 0$ s.t.

$\forall\ n > n_0, cg(n) \leq |f(n)|$

$f = o(g)$

f grows no faster than g

$\forall c > 0, \exists\ n_0$ s.t.

$\forall\ n > n_0, |f(n)| \leq cg(n)$

Asymptotic Relationships

$$\begin{aligned} f = O(g), f = \Omega(g) &\iff f = \Theta(g) \\ f = O(g) &\iff g = \Omega(f) \\ f = o(g) &\implies f = O(g) \\ \log n = O(n^\epsilon)\ \forall\ \epsilon > 0 & \end{aligned}$$

2 Geometric Series

Series where each successive term is a constant ratio of the prior.

$$S_n = \sum_{i=0}^{n} ar^i$$

Useful things:

- When $|r| < 1$ we have $\sum_{i=0}^{\infty} ar^i = \frac{a}{1-r}$
- For $|r| > 1$, $\sum_{i=0}^{n} f(n) r^i = O(f(n))$
- $1 + \frac{1}{2} + \frac{1}{4} + \cdots = 2$
- When $r > 1$, we have $S_n = \sum_{i=0}^{n} ar^i \approx ar^n$

3 Logarithm Rules

$$\begin{aligned} \log_a 1 &= 0 \\ \log_a a &= 1 \\ \log_a a^n &= n \\ \log_a (xy) &= \log_a x + \log b \\ \log_a (x/y) &= \log_a x - \log_a y \\ \log_a x^n &= n \log_a x \\ \log_a x &= \log_b x / \log_b a \end{aligned}$$

[1] https://web.mit.edu/broder/Public/asymptotics-cheatsheet.pdf

4 Master Theorem

Given a recurrence

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq 1 \\ a \cdot T\left(\frac{n}{b}\right) + f(n) & \text{otherwise} \end{cases}$$

where $a \geq 1$, $b > 1$, and $f(n)$ is asymptotically positive we can bound the work with:

1: $f(n)$ polynomially smaller than $n^{\log_b a}$ $\implies \Theta(n^{\log_b a})$

2: $f(n)$ polynomially equal to $n^{\log_b a}$ $\implies \Theta(f(n) \log n)$

3: $f(n)$ polynomially larger than $n^{\log_b a}$ $\implies \Theta(f(n))$

To prove one of the three cases you need to show that:

1: $\exists\ \epsilon > 0$ s.t. $f(n) = O\left(n^{\log_b (a - \epsilon)}\right)$

2: $f(n) = \Theta\left(n^{\log_b a}\right)$

3: $\exists\ \epsilon > 0$ s.t. $f(n) = \Omega\left(n^{\log_b (a + \epsilon)}\right)$

and $\exists c \leq 1$ s.t. $af\left(\frac{n}{b}\right) \leq cf(n)$

5 Algorithms

5.1 Matrix Multiplication

$A \cdot B = C$ where

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

Naive is $O(n^3)$

Strassen's Algorithm is $O\left(n^{\log_2 7}\right) = O\left(n^{2.81}\right)$

6 Data Structures

$O(1)$ to lookup in a map or set

$O(n)$ to scan a list or array of size n