Cheat Sheet Efficient Algorithms

Paul

February 12, 2025

1 General Stuff

1.1 Asymptotic Notation

The set of functions that asymptotically grow not faster than $g(n)$ are:

$$\mathcal{O}\left(g\left(n\right)\right) = \{f : \mathbb{N} \to \mathbb{R}^+ \mid \exists c, n_0 \in \mathbb{R}^+ \text{ such that } \forall n \geq n_0 : \boldsymbol{f(n) \leq c \cdot g(n)}\}$$

The set of functions that asymptotically grow not slower than $g(n)$ are:

$$\Omega\left(g\left(n\right)\right) = \{f : \mathbb{N} \to \mathbb{R}^+ \mid \exists c, n_0 \in \mathbb{R}^+ \text{ such that } \forall n \geq n_0 : \boldsymbol{f(n) \geq c \cdot g(n)}\}$$

The set of functions that asymptotically grow at the same rate as $g(n)$ are:

$$\Theta\left(g\left(n\right)\right) = \mathcal{O}\left(g\left(n\right)\right) \cap \Omega\left(g\left(n\right)\right)$$

2 Recursion

2.1 Master Theorem

Let $a \geq 1$, $b > 1$, $\epsilon > 0$ then

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

then:

- If $f(n) = \mathcal{O}\left(n^{\log_b(a) - \boldsymbol{\epsilon}}\right)$ then $T(n) = \Theta\left(n^{\log_b a}\right)$
- If $f(n) = \Theta\left(n^{\log_b(a) \cdot \boldsymbol{\log^k n}}\right)$ then $T(n) = \Theta\left(n^{\log_b a} \log^{k+1} n\right)$
- If $f(n) = \Omega\left(n^{\log_b(a) + \boldsymbol{\epsilon}}\right)$ and $af\left(\frac{n}{b}\right) \leq cf(n)$ for some $c < 1$ and sufficiently large n then $T(n) = \Theta\left(f\left(n\right)\right)$

2.2 Proof by Induction for Recursion

To show that $f(n) = \Theta\left(g\left(n\right)\right)$ we perform the following algorithm:

- Guess that $f(n) = \mathcal{O}\left(g\left(n\right)\right)$ and $f(n) = \Omega\left(g\left(n\right)\right)$
- Prove that $f(n) = \mathcal{O}\left(g\left(n\right)\right)$ and $f(n) = \Omega\left(g\left(n\right)\right)$ by induction
- The inductive hypothesis is that $T(n) = \mathcal{O}\left(g\left(n\right)\right)$, i.e. $T(n) \leq c \cdot g(n)$ for some constant c and sufficiently large n
- Analogously for $\Omega\left(g\left(n\right)\right)$

2.3 Linear Homogeneous Recurrence Relations

Given the recursion $a_n = \sum_{i=1}^{k} c_i a_{n-i}$ we can solve it by finding the characteristic polynomial $p(x) = x^k - \sum_{i=1}^{k} c_i x^{k-i}$ and then solving it for the roots $x_1, x_2, \ldots, x_k$. The general solution is then $a_n = \sum_{i=1}^{k} \alpha_i x_i^n$ where the α_i are determined by the initial conditions. For roots x_i with multiplicity m_i we have to include terms of the form $n^j x_i^n$ for $j = 0, 1, \ldots, m_i - 1$ in the general solution. Example:

$$\begin{aligned} a_n &= 3a_{n-2} - 2a_{n-3} \\ \mathcal{X}(\lambda) &= \lambda^3 - 3\lambda + 2 \\ &= (\lambda+1)(\lambda+1)(\lambda-2) \end{aligned}$$

With the initial conditions $a_0 = 3, a_1 = 2, a_2 = 1$ we get:

$$a_n = \alpha\,(-1)^n + n \cdot \beta\,(-1)^n + \gamma 2^n$$

Solving for the constants:

$$\begin{aligned} \alpha + 0 + \gamma &= 3 & \Rightarrow \alpha &= 3 - \gamma \\ -\alpha - \beta + 2\gamma &= 2 & \Rightarrow \beta &= 3\gamma - 5 \\ \alpha + 2\beta + 4\gamma &= 11 & \Rightarrow \gamma &= 2 \end{aligned}$$

And therefore the solution is:

$$\begin{aligned} a_n &= (-1)^n + n\,(-1)^n + 2 \cdot 2^n \\ &= (-1)^n\,(1+n) + 2^{n+1} \end{aligned}$$

2.4 Inhomogeneous Recurrence Relations

Given the recursion $a_n = \sum_{i=1}^{k} c_i a_{n-i} + f(n)$ we can solve it by transforming it into a homogeneous recursion. To do this we iteratively substitute the recursion into itself until we get a homogeneous recursion. For example:

$$\begin{aligned} a_n &= a_{n-1} + n^2 \\ a_{n-1} &= a_{n-2} + (n-1)^2 = a_{n-2} + n^2 - 2n + 1 \end{aligned}$$

Adding and subtracting the two equations we get:

$$\begin{aligned} a_n &= 2a_{n-1} + n^2 - a_{n-1} \\ &= 2a_{n-1} + n^2 - \big(a_{n-2} + n^2 - 2n + 1\big) \\ &= 2a_{n-1} - a_{n-2} + 2n - 1 \end{aligned}$$

Similarly for a_{n-1}

$$\begin{aligned} a_{n-1} &= a_{n-2} + (n-1)^2 \\ &= a_{n-2} + n^2 - 2n + 1 \\ a_{n-2} &= a_{n-3} + (n-2)^2 \\ &= a_{n-3} + n^2 - 4n + 4 \\ a_{n-1} &= 2a_{n-2} + \big(n^2 - 2n + 1\big) - \big(a_{n-3} + n^2 - 4n + 4\big) \\ &= 2a_{n-2} - a_{n-3} + 2n - 3 \end{aligned}$$

And going back to the original equation:

$$\begin{aligned} a_n &= 3a_{n-1} - a_{n-2} + 2n - 1 - (2a_{n-2} - a_{n-3} + 2n - 3) \\ &= 3a_{n-1} - 3a_{n-2} + a_{n-3} + 2 \end{aligned}$$

We have to repeat this process until we get a homogeneous equation.

$$a_{n-1} = 3a_{n-2} - 3a_{n-3} + a_{n-4} + 2$$

And finally:

$$\begin{aligned} a_n &= 4a_{n-1} - 3a_{n-2} + a_{n-3} + 2 - a_{n-1} \\ &= 4a_{n-1} - 3a_{n-2} + a_{n-3} + 2 - (3a_{n-2} - 3a_{n-3} + a_{n-4} + 2) \\ &= 4a_{n-1} - 6a_{n-2} + 4a_{n-3} - a_{n-4} \end{aligned}$$

2.5 Generating Functions