Reading 6 - 4.5 The Master Method for Solving Recurrences

Goals

- Understand:
  - What form of recurrences this method can be applied to
  - What the three cases are and how you determine into what case a certain recurrence falls
  - What does "polynomially larger" mean?

Master Method

The **master method** is a "cookbook" method for solving recurrences. It *HAS* to be in the form:

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

Let's extract the variables:

- $a \geq 1$
- $b > 1$
- $f(n)$: an **asymptotically positive** function

For each run recursion, the argument to $T(n)$ is divided by b.

Typically, we want $a < b$ which means we have a **good** runtime. Otherwise, if $a > b$ we have a **bad** runtime.

$T(n)$ has the following asymptotic bounds or **cases**:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \theta(n^{\log_b a})$
2. If $f(n) = \theta(n^{\log_b a})$, then $T(n) = \theta(n^{\log_b a} \log n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and sufficiently large n, then $T(n) = \theta(f(n))$

How to Use the Master Method

Say we have the recurrence:

$$T(n) = 9T(\frac{n}{3}) + n$$

We define $a = 9, b = 3, f(n) = n$. Let's plug those into our **case** test:

$$n^{\log_b a}$$

$$n^{\log_3 9}$$

$$n^2$$

From there, we need to get the epsilon. To do that, we need to gather $f(n)$ and $n^{\log_b a}$:

- $f(n) = n = n^1$
- $n^{\log_b a} = n^2$

To get the epsilon, we subtract the exponent from $n^{\log_b a}$ from the exponent of $f(n)$:

$$\epsilon = 2 - 1 = 1$$

Since $\epsilon > 0$, we can use **case 1** to get the solution:

$$T(n) = \theta(n^{\log_3 9}) = \theta(n^2)$$

The solution to the recurrence $T(n) = 9T(\frac{n}{3}) + n$ is $\theta(n^2)$.

Polynomially Larger

Generally, a function is **polynomially larger** if it grows faster than another by at least n^ϵ for some constant $\epsilon > 0$.

What does this look like? For example, if $n^{\log_b a} = n^2$ and $f(n) = n^3$, then $f(n)$ is polynomially larger than n^2 because $n^3 = n^{2+1} = n^{\log_b a+1}$.

However, if $n^{\log_b a} = n$ and $f(n) = n \lg n$, then $n \lg n$ is NOT polynomially larger than n because in $n \lg n$, multiplying n by $\lg n$ results in a slower increase than n.