Last time we talked about Polynomial Multiplication.  
Today we will talk about Matrix Multiplication and the Master Theorem. 

Task: Multiply $2 \times n \times n$ matrices

Matrices: $A \cdot B = C$

$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$

$n^2$ positions in each of the above matrices.

Method #1: 2-hand method (school method)

- $n$ multiplications and $(n - 1)$ additions for each $C_{ij}$

Complexity of Run-Time: $O(n^3)$

Example: a $2 \times 2$ matrix has 8 multiplications and 4 additions.

Method #2:

- $x_1 = (a_{11} + a_{22}) \cdot (b_{11} + b_{22})$
- $x_2 = (a_{21} + a_{22}) \cdot b_{11}$
- $x_3 = a_{11} \cdot (b_{12} - b_{22})$
- $x_4 = a_{22} \cdot (b_{21} - b_{11})$
- $x_5 = (a_{11} + a_{12}) \cdot b_{22}$
- $x_6 = (a_{21} - a_{11}) \cdot (b_{11} + b_{12})$
- $x_7 = (a_{12} - a_{22}) \cdot (b_{21} + b_{22})$

$c_{11} = x_1 + x_4 - x_5 + x_7$
$c_{12} = x_3 + x_5$
$c_{21} = x_2 + x_4$
$c_{22} = x_1 + x_3 - x_2 + x_6$

We will not be required to memorize this algorithm. It is for show purposes only.

Analysis:

- Total Multiplications: 7
- Total Additions: 18

On a small $2 \times 2$ matrix this method isn’t as good as the other “school method.” However, for larger matrices it is useful.

Matrices can be reduced into blocks to solve:
\[
\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}
\]

\[C_{11} = A_{11}B_{11} + A_{12}B_{21}\]

\[
\ldots
\]

This method can be solved using recursion.

\[
T(n) = \begin{cases} 7T\left(\frac{n}{2}\right) + cn^2 & n > 1 \\ \alpha & n = 1 \end{cases}
\]

This is the general representation for these recurrences:

\[
T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + d(n) & n > 1 \\ \alpha & n = 1 \end{cases}
\]

\[a, b, \alpha \text{ are constants}\]

\[= a \cdot T\left(\frac{n}{b}\right) + d(n)\]

\[= a\left(a \cdot T\left(\frac{n}{b^2}\right) + d\left(\frac{n}{b}\right)\right) + d(n)\]

\[= a^2 \cdot T\left(\frac{n}{b^2}\right) + d(n) + a \cdot d\left(\frac{n}{b}\right)\]

\[\ldots\]

\[i^{th} = a^i \cdot T\left(\frac{n}{b^i}\right) + d(n) + a \cdot d\left(\frac{n}{b}\right) + \ldots + a^{i-1} \cdot d\left(\frac{n}{b^i - 1}\right)\]

Note: Assumption that will hold for some functions \(d(n)\):

\[d(n) = c \cdot f(n) \text{ where } f(n) \text{ is multiplicative}\]

[a function is multiplicative if \(f(x \cdot y) = f(x) \cdot f(y)\)]

\[= a^i \cdot T\left(\frac{n}{b^i}\right) + c \cdot f(n) \left(1 + a \cdot f\left(\frac{1}{b}\right) + a^2 \cdot f^2\left(\frac{1}{b}\right) + \ldots + a^{i-1} \cdot f^{i-1}\left(\frac{1}{b}\right)\right)\]

This is now a geometric series:

\[
\sum_{j=0}^{i-1} \left(a \cdot f\left(\frac{1}{b}\right)\right)^j = \frac{x^i - 1}{x - 1} \quad x \neq 1
\]

\[= a^i \cdot T\left(\frac{n}{b^i}\right) + c \cdot f(n) \cdot \frac{a^i \cdot f\left(\frac{1}{b}\right)^{i-1}}{a \cdot f\left(\frac{1}{b}\right)^{i-1}}\]

\[\frac{n}{b^i} = 1 \Rightarrow n = b^i, i = \log_b n\]

\[= a^{\log_b n} \cdot \alpha + c \cdot f(n) \cdot \frac{a^{\log_b n}, f\left(\frac{1}{b}\right)^{\log_b n} - 1}{f\left(\frac{1}{b}\right)^{\log_b n} - 1} = f^{\log_b n} \left(\frac{1}{b}\right) = f\left(\frac{1}{b^{\log_b n}}\right) = f\left(\frac{1}{n}\right) = \frac{1}{f(n)}\]

\[f\left(\frac{1}{b}\right) = \frac{f(1)}{f(b)} = \frac{1}{f(b)}\]
= a^{log_b a} \cdot \alpha + c \cdot \frac{a^{log_b a} \cdot f(n)}{n - 1}

Case 1: a < f(b) complexity: T(n) = O(f(n))
Case 2: a > f(b) complexity: T(n) = O(n^{log_b a})
Case 3: a = f(b) complexity: T(n) = O(f(n) \cdot log_b n)

Above we derived the Master Theorem, now for the formal definition:

**Master Theorem:**
If T(n) is recursively defined as

\[ T(n) = \begin{cases} 
   a \cdot T \left( \frac{n}{b} \right) + d(n) & n > 1 \\
   \alpha & n = 1 
\end{cases} \]

Where a, b, \alpha are constants and d(n) = c \cdot f(n) where c is constant and f(n) is multiplicative then

\[ T(n) = \begin{cases} 
   O(f(n)) & a < f(b) \\
   O(f(n) \cdot \log n) & a = f(b) \\
   O(n^{log_b a}) & a > f(b) 
\end{cases} \]

**Memorize the Master Theorem. Just the theorem, not the derivation of it**

Back to matrices:

\[ T(n) = 7 \cdot T \left( \frac{n}{2} \right) + c \cdot n^2 \]

a = 7
b = 2
f(n) = n^2 \Rightarrow a > f(b)
f(b) = 4

\[ T(n) = O(n^{log_2 7}) \]
\[ T(n) = O(n^{log_2 8}) \]

Merge sort with Master Theorem (it is easy):

\[ T(n) = 2 \cdot T \left( \frac{n}{2} \right) + n \]
a = 2
b = 2
= O(n \cdot log n)

It would be good to read chapter 4 in the book to learn more about this.

Next time we will formally introduce \( O() \) and the growth of functions.