Matrix Multiplication

The algorithm for this course is simply matrix multiplication. This is a scalable algorithm and can be used to measure the viability of any parallel programming paradigm. If you cannot make matrix multiplication scale in a parallel programming paradigm then said paradigm may not be viable.

For two matrices \( A \) and \( B \) to be multiplied together they must have the following characteristics. The number of columns in matrix \( A \) must be identical to the number of rows in matrix \( B \). The resultant product of those to matrices, \( C \), will have the same number of rows that matrix \( A \) has and the same number of columns as matrix \( B \). Therefore,

\[
\begin{align*}
\text{number of columns}_A &= \text{number of rows}_B \\
\text{number of rows}_A &= \text{number of rows}_C \\
\text{number of columns}_B &= \text{number of columns}_C
\end{align*}
\]

The simple representation of any given row of matrix \( A \) multiplied by any given column of matrix \( B \) gives the resultant element of matrix \( C \). Mathematically this becomes:

\[
C_{i,j} = \sum_{k=K_{\text{start}}}^{K_{\text{limit}}} A_{i,k} \times B_{k,j}
\]  

Where \( k \) is the summation index over the length of the row in \( A \) or the length of the column of \( B \). \( K_{\text{start}} \) is 0 (zero) for a C program and 1 for FORTRAN. \( K_{\text{limit}} \) is the lexical index representing the length of the row in \( A \) (for a C program it is \( \text{number of columns}_A - 1 \), for FORTRAN it is \( \text{number of columns}_A \)). \( K_{\text{limit}} \) is also the lexical index representing length of the column in \( B \) (for a C program it is \( \text{number of rows}_B - 1 \), for FORTRAN it is \( \text{number of rows}_B \)).

This should make sense because the “width” of \( A \) must equal the “height” of \( B \), in order to multiply the two matrices.

Again Equation 4 is for each element of the \( C \) product or result matrix. How many elements does \( C_{i,j} \) have? It has \( \text{number of rows}_C \times \text{number of columns}_C \). From Equations 2 and 3 this is equivalent to: \( \text{number of rows}_A \times \text{number of columns}_B \)

This leads us directly to the simple loop or dot product (“\( \text{ddot} \)” form of matrix multiplication:

```cpp
for (i=0; i<\text{number of rows}_C; i++) {
    for (j=0; j<\text{number of columns}_C; j++) {
        for (k=0; k<\text{number of columns}_A; k++) {
            C[i][j] += A[i][k] \times B[k][j];
        }
    }
}
```
