CS 331 Project #3

Jong Woo Lee

12/10/14

**Abstract:**

Project #3 requires us to implement both the iterative version of matrix multiplication and the algorithm created by Volker Strassen (Strassen's Algorithm). We want to be able to compare the two algorithm and analyze the strengths and weaknesses that both algorithms have. If tests are successfully ran, we will be able to see that Strassen's algorithm would only slightly faster than the standard algorithm for the matrix multiplication. The idea of the Strassen's method is to reduce the number of recursive calls. Strassen's algorithm has a time complexity of: T(N) = 7T(N/2) + O(N2)

**Algorithm Discussion/Presentation:**

Starting with the classical approach for matrix multiplication, we have matrices A,B and where C = A \* B. The classical approach has three nested loops and will output a cost of Θ(n3).

Now with Strassen's algorithm, we have two matrices A and B. In those matrices, we will have sub-matrices in each matrices. Strassen's algorithm has the key idea where we multiply matrices using 7 scalar multiplies instead of 8

Matrix A contains values: A,B,C,D as the sub matrices of A

Matrix B contains values: E,F,G,H as the sub matrices of B

The seven quadrant multiplications is computed recursively and the computational cost of addition and subtraction will be Θ(n2) --> thus making the recurrence equation: F(n) = 7F(n/2) + Θ(n2)

In the Strassen's method that I have implemented, we have the code:

**if** (number == 1) {

c[0][0] = a[0][0] \* b[0][0];

}

Strassen's algorithm has a smaller complexity but a bigger Big-O constant and the function call will head all the way down to 1 element.

The method will divide the matrices to sub-matrices of size N/2 x N/2 but the four matrices of result will be calculated as follows:

M1 = A(F-H)

M2 = (A + B)H

M3 = (C + D)E

M4 = D(G - E)

M5 = (A + D)(E + H)

M6 = (B - D)(G + H)

M7 = (A - C)(E + F)

We iterate through the division and the matrices will eventually form numbers.

**Complexity Analysis:**

Complexity of Strassen's Algorithm & Matrix Multiplication:

(http://www.geeksforgeeks.org/strassens-matrix-multiplication/)

**Test Case Design:**

Matrix Multiplication iterative algorithm has nested for loops where they all go from 1 to n so the best, average, and worst case will all output the same O(n \* n \* n) = O(n3).

Strassen's algorithm is not used for practical applications for the following reasons:

1) The constants that are used in the algorithm are high and do not work well with typical applications. They are better when the number/size is larger.

2) There are better methods to use for sparse matrices.

3) The recursion for sub-matrices take up extra space\

4) The limited precision of the computer arithmetic makes non-integer values to compile larger errors in Strassen's algorithm than the regular approach.

**Experiment:**

In practice, it is difficult to match the execution time complexity to the theoretical complexity when implementing because of the time complexity and the space complexity. It is difficult to handle the time complexity of this algorithm and because of this reason we can't see much further improvement. The improvement of the run time complexity of the algorithm would mean an improvement in the space complexity as well.

Iterative Algorithm

|  |  |
| --- | --- |
| Size | Execution Time (ms) |
| 2 | 0 |
| 4 | 0 |
| 8 | 0 |
| 16 | 1 |
| 32 | 1 |
| 64 | 2 |
| 128 | 9 |
| 256 | 1219 |
| 512 | 1560 |
| 1024 | 15320 |

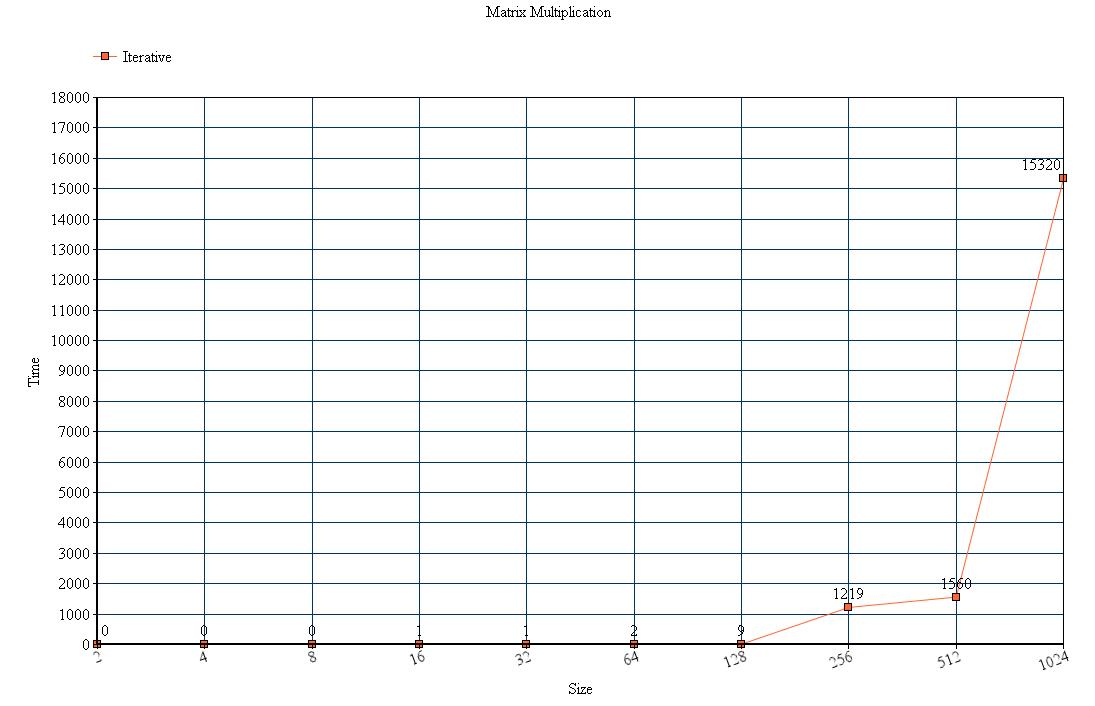
Divide & Conquer Algorithm

|  |  |
| --- | --- |
| Size | Execution Time (ms) |
| 2 | 0 |
| 4 | 0 |
| 8 | 1 |
| 16 | 7 |
| 32 | 53 |
| 64 | 432 |
| 128 | 1786 |
| 256 | 14661 |
| 512 | 123456 |
| 1024 | 985235 |

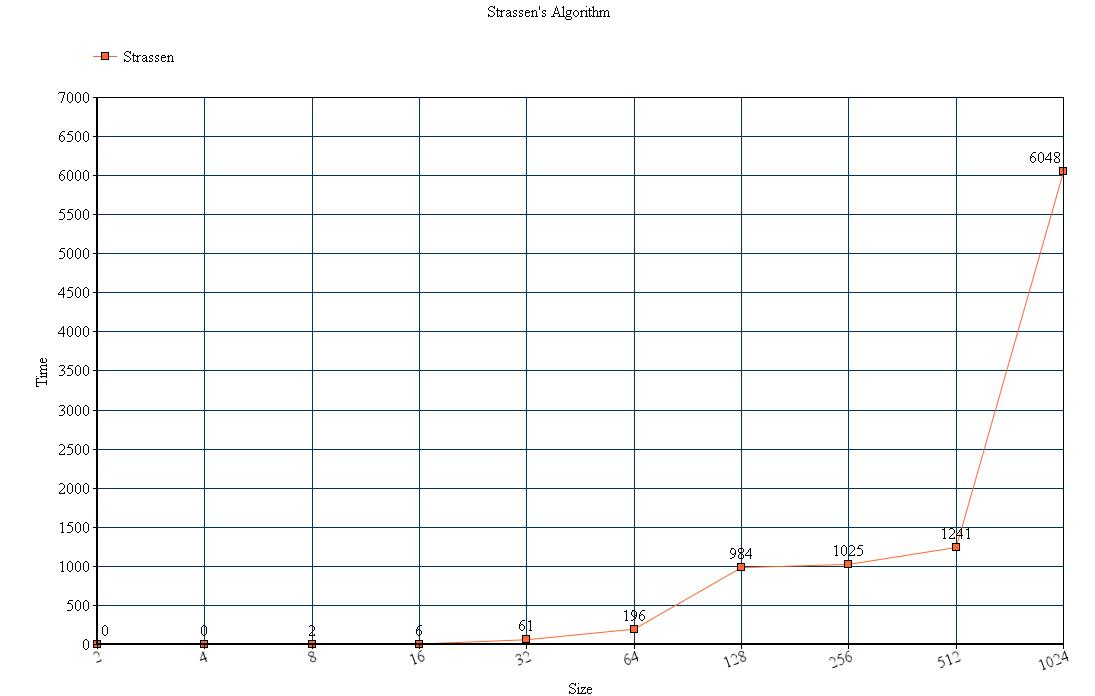
Strassen's Algorithm

|  |  |
| --- | --- |
| Size | Execution Time (ms) |
| 2 | 0 |
| 4 | 0 |
| 8 | 2 |
| 16 | 6 |
| 32 | 61 |
| 64 | 196 |
| 128 | 984 |
| 256 | 1025 |
| 512 | 1241 |
| 1024 | 6048 |

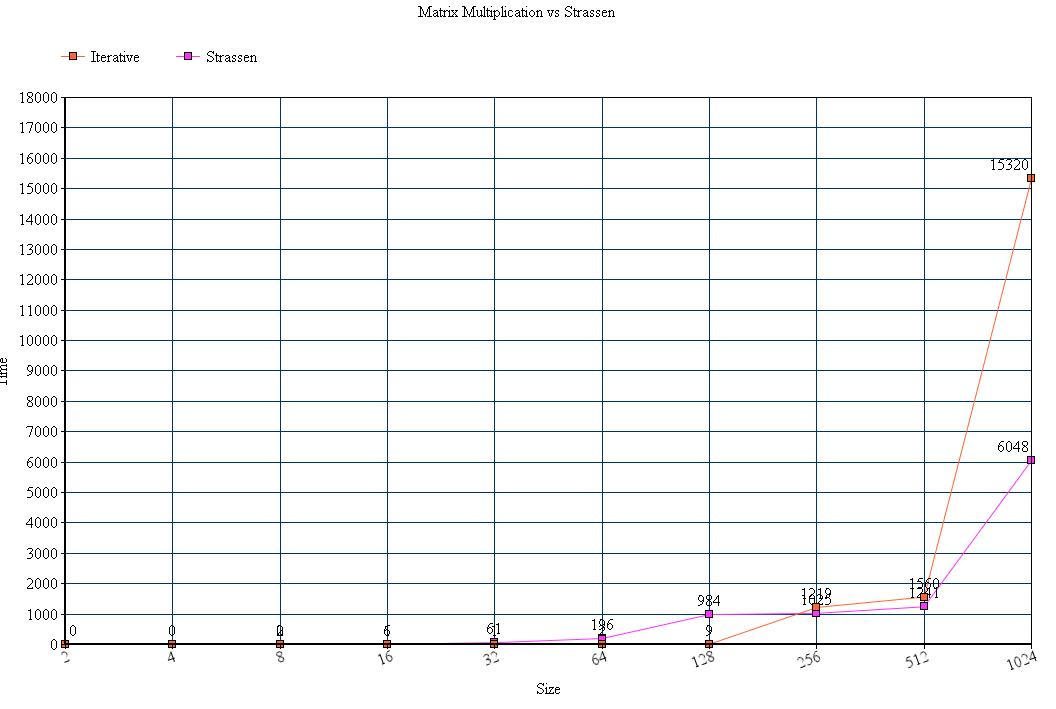
Graph (Matrix Multiplication)



Graph (Strassen's Algorithm)



Graph (Matrix Multiplication vs Strassen's Algorithm)



The two intersect at around size 240 - 256.

**Conclusion:**

In conclusion, some of the tested data need to be investigated in the terms of execution time because it can widely change. In practical terms, it is better to use matrix multiplication because we n x n arrays. Strassen's algorithm is faster but not that much faster than the original matrix multiplication. For most part, n is going to be for small values so the general algorithm is just a better choice. As stated above, there are better methods for sparse matrices, the constants used in Strassen's method is high, the sub-matrices in recursion take extra space, and because of the limited precision of the computer arithmetic, it will output some errors.