Project 1 – CSC 501

In Chapter 4 of the book we got introduced to Strassen's algorithm for matrix multiplication. In that section the book showed the following three methods to multiple two square matrices: Nested Loops, Recursion, and Strassen's. Using either C, Python, or Java you need to implement each of these algorithms or cite code you got from the Internet. The implementations need to be able to handle any value of n from 2 to 1,000. Note the book assumed a power of 2 and exercise 4.2-3 asks you to make Strassen's algorithm more general. You will need a version that runs on integers and floats.

Using these algorithms you will gather time information to see how the implementations match the expected O() time bounds of each. Getting good timing values for n being small will likely require having the specific algorithm run multiple times on an n that size. So for example if n = 2, you might have to start the timer, then run the algorithm 1,000 times, then stop the timer and divide the result by 1,000. So that our measurements are as accurate as possible you should run all your tests on the same computer with roughly the same applications running in the background each time. Additionally, you must run each timing test 10 times to get some idea of how much the time can vary by due to background processes.

Experiment Design

Algorithm: Nested Loops, Recursion, and Strassen's
Data Types: Integer and Float
Matrix Size: 10x10, 100x100, and 1000x1000 (*Note I stuck with powers of 10*)
Total number of treatments: 18 (3 Algorithms * 2 Data Types * 3 Matrix Sizes)
For each treatment you will do 10 trials.

NOTE: If we wanted to be really through, we would also run comparisons for Power of 2 as well.

What to Submit:

1. The code for each algorithm.

2. Graphs that show the average and standard deviation for each treatment. For some helpful guidance on picking the right graph see here: [How to Choose the Right Chart for Your Data (infogram.com)]

Now an individual treatment is just two values: average and standard deviation. So you will need to figure out what would be the best way to group the treatments. This should connect with how you are comparing the values. For example, if it is all results within an algorithm, then group that way or if you are comparing the values based on matrix size group that way.

3. We know the asymptotically tight bound for each algorithm. Choose one data type for each algorithm and show mathematically how your results either match or don't match the expected bounds of each algorithm.

4. We varied the data type, did this affect performance within the same algorithm? Use a t-test to determine if two values are the same or different (see below on an explanation on how to compute and use a t-test). Include a table of the results and write up a short summary for each data type (2-5 sentences each – length depends on how consistent the results are).

Computing the T-Test Value

Figure 1 shows the actual formula for computing the ***T-test*** value of two averages. Without going into great detail the ***t-test*** functions by first computing the standard deviation (s_1 and s_2) for each average (x_1 and x_2). The belief in this accuracy measure (standard deviation) is based on the number of observations (n_1 and n_2 - number of values that make up each average). The more observations, the more likely the accuracy measure is correct. The overall accuracy measure (δ_{xx}) is then used to adjust the absolute difference between the two averages.

Thus, the smaller the difference or the larger the inaccuracy in the averages the smaller the value computed by the ***t-test***. The smaller the value from the t-test the more likely the two averages are similar.

$$t = \frac{\overline{x}_1 - \overline{x}_2}{\delta_{xx}}, \quad \delta_{xx} = \sqrt{\left(\frac{(n_1)(s_1)^2 + (n_2)(s_2)^2}{(n_1 - 1) + (n_2 - 1)}\right)} * \sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}$$

Figure 1: Formula for Comparison of Two Observed Means

After you get the ***t*** value to determine if they are the same or different there is a table in the back of most Stats books. You first need to know the df or degrees of freedom, which is determined by the smallest ***n-1*** of the two samples, so 9 in our case. After that we are usually looking at about 95% confidence or higher. So using the table I can see that a ***t*** value of 2.262 or higher means the values are likely different. A value smaller than that and they are likely similar, close means it could be either.