Finally, the last memory-access-related optimization was reducing the overhead of accessing array elements within the innermost loop. Namely, by holding the value of $\text{matrix}_a[i, k]$ in a temporary variable, we avoid multiple look-ups which can tack on significant overhead.

Strassen Optimizations: Another major optimization was done on the calculation of the Strassen subproblems $P_1, \ldots, P_7$ by concurrently calculating all of these values through the use of multi-threading, using the ThreadPoolExecutor class from the futures module.

Finally, the last obvious optimization was the use of a "cross-over point"–the optimal base case for the recursion of our Strassen algorithm (the cross-over point specified in the assignment). At this crossover point, recursing to one level above or below, then calculating the base case using the simple matrix multiplication algorithm would take longer. We provide a detailed analysis of how we found this crossover point in the section below.

Testing: Along the way, we tested our algorithm on smaller matrices using 1s and 0s. We initially tested on matrices with randomly generated values between 0 and 2 but the actual matrix values did not seem to affect the runtime or crossover points we found in any substantial way when compared to matrices with 0s and 1s. Consequently, we thought it would be simpler to check if our algorithm was correct using Wolfram Alpha's matrix calculator if we used 0s and 1s.

2 Crossover Point Analysis

2.1 Analytical/Theoretical Analysis

We assume that standard operations, such as addition, subtraction, multiplication, and division, all take 1 unit of time to complete. With this assumption, we can now compute the total work done under the naive matrix multiplication algorithm and Strassen's algorithm for $n \times n$ matrices. Denote the former as $c_N(n)$ and the latter as $c_S(n)$.

Work Done in Standard Matrix Multiplication: Suppose we have 2 $n \times n$ matrices A and B. Then, the product $C = AB$ must have n^2 entries. Under the naive matrix multiplication algorithm, we perform n multiplications and $n - 1$ addition operations (intuitively, the second fact follows from how summing, say 2 numbers, involves 1 addition operation). Note that we do this for each of the n^2 elements in the product matrix, so for any given element the total work done is $n + n - 1 = 2n - 1$. Then, our total work under the naive algorithm is as follows:

$$\begin{aligned} c_N(n) &= n^2(2n-1) \\ &= 2n^3 - n^2 \end{aligned}$$

Work Done in Strassen's Algorithm: First, recall that the number of steps or work done under Strassen's algorithm for an *even* value of n is given by $T(n) = 7T(n/2) + O(n^2)$; the first term simply follows from the algorithm splitting the size n problem into 7 size $n/2$ problems, and the second term is the work done at each level of recursion. Now, we need to pinpoint the exact form of the $O(n^2)$ to determine $c_S(n)$. For clarity, this recurrence assumes that certain operations such as splicing and computing $n/2$ are both $O(1)$ operations. Recall that we split the 2 matrices M_1, M_2 into sub-matrices A, B, C, D and E, F, G, H each of size $n/2$. In computing the following subproblems and combining the subproblems of size $n/2$, observe we perform 18 additions/subtractions:

$$\begin{aligned} P_1 &= A(F-H) \\ P_2 &= (A+B)H \\ P_3 &= (C+D)E \\ P_4 &= D(G-E) \\ P_5 &= (A+D)(E+H) \\ P_6 &= (B-D)(G+H) \\ P_7 &= (A-C)(E+F) \end{aligned}$$

Here, we have 10 multiplications/additions on matrices of size $n/2$. Next, in the combine step of Strassen's, we perform an additional 8 additions/subtractions:

$$\begin{aligned} &Q_1 : P_5 + P_4 - P_2 + P_6 \\ &Q_2 : P_1 + P_2 \\ &Q_3 : P_3 + P_4 \\ &Q_4 : P_1 - P_3 + P_5 + P_7 \end{aligned}$$

Hence, we have a total of 18 additions/subtractions on matrices of size $n/2$ at each level of recursion. Then, we can define our total work done under Strassen's algorithm as follows:

$$\begin{aligned} c_S(n) &= 7c_S(n/2) + 18(\frac{n}{2})^2 \\ &= 7c_S(n/2) + \frac{9}{2}n^2 \end{aligned}$$

Note that $c_S(1) = 1$ here since it denotes simple integer multiplication.

Analytical Crossover Point: To determine the crossover point analytically, we want some n_A such that $c_N(n_A) = c_S(n_A)$. We can substitute for each respective equation:

$$2n_A^3 - n_A^2 = 7c_S(\frac{n_A}{2}) + \frac{9}{2}n_A^2$$

Suppose that indeed $c_N(n_A) = c_S(n_A)$. Then we can further simplify as follows:

$$\begin{aligned}
2n_A^3 - n_A^2 &= 7c_S(\frac{n_A}{2}) + \frac{9}{2}n_A^2 \\
2n_A^3 - n_A^2 &= 7(\frac{2n_A^3}{8} - \frac{n_A^2}{4}) + \frac{9}{2}n_A^2 \\
\frac{1}{4}n_A^3 + \frac{3}{4}n_A^2 &= \frac{9}{2}n_A^2 \\
n_A^3 - 15n_A^2 &= 0 \\
n_A^2(n_A - 15) &= 0 \\
\implies n_A &= 15
\end{aligned}$$

Hence, we find an analytical cross-over point of $n_A = 15$. This tells us that, for matrices with dimension $n < 15$, the work done to compute the product of $n \times n$ matrices is less than that under Strassen's algorithm. Furthermore, this cross-over point relies on the assumption that addition/subtraction are indeed 1 unit of work relative to other operations which may or may not be necessarily true. The other major assumption here is that n is even. We handle the odd case now.

Addendum (Non-Even n): Recall that the recurrence relation for Strassen's algorithm is given by $T(n) = 7T(n) + O(n^2)$. This recurrence relation makes the assumption that n is *even.* Now suppose that n is in fact odd. We now find the cross over-point in this case. To do this, we need to *pad* the matrix. There are several approaches to this, but the implementation simply pads to increase the dimension by 1, thereby converting the sub-matrix from an odd-numbered dimension to an even-numbered dimension. In other words, the problems are divided into sub-problems of size $\frac{n+1}{2}$. Hence, we find the following function pinpointing the work done under Strassen's in the case of an odd value of n:

$$\begin{aligned}
c_S(n) &= 7c_S(\frac{n+1}{2}) + 18(\frac{n+1}{2})^2 \\
&= 7c_S(\frac{n+1}{2}) + \frac{9}{2}(n+1)^2
\end{aligned}$$

Now, we proceed as before in determining the analytical cross-over point $n_{A_{odd}}$. For sake of notation, let $n_* = n_{A_{odd}}$ Suppose that $c_N(n_*) = c_S(n_*)$.

$$\begin{aligned}
&2n_*^3 - n_*^2 = 7c_S(\frac{n_*+1}{2}) + \frac{9}{2}n_*^2 \\
&2n_*^3 - n_*^2 = 7(2(\frac{n_*+1}{2})^3 - (\frac{n_*+1}{2})^2) + \frac{9}{2}(n_*+1)^2 \\
&\frac{1}{4}n_*^3 - 9n_*^2 - \frac{43n_*}{4} - \frac{9}{2} = 0 \\
&\implies n_* = 37.17 \approx 37
\end{aligned}$$