Problem A (CLRS 28.2-2)

How would you modify Strassen’s algorithm to multiply \( n \times n \) matrices in which \( n \) is not an exact power of 2? Show that the resulting algorithm runs in time \( \Theta(n^{\lg 7}) \).

Strassen’s algorithm can be applied to \( n \times n \) matrix multiplications where \( n \) is not an exact power of 2 by padding the operands with 0’s. Let \( m = 2^k \) such that \( 2^{k-1} < n < 2^k \) \( (m \text{ equals } 2^{\lceil \lg n \rceil}) \). Create \( m \times m \) matrices \( A’ \) and \( B’ \) by padding \( A \) and \( B \) respectively.

Applying Strassen’s algorithm, the resulting matrices \( C’ \), \( A’ \) and \( B’ \) appear as follows, where \( C’ \) is the matrix product of \( A’ \) and \( B’ \):

\[
C' = \begin{bmatrix} C & 0 \\ 0 & 0 \end{bmatrix} \quad A' = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} \quad B' = \begin{bmatrix} B & 0 \\ 0 & 0 \end{bmatrix}
\]

To obtain the product, we simply extract the matrix \( C \) from \( C’ \).

The runtime for this method is \( \Theta(m^{\lg 7}) \). Since \( 2^{k-1} < n \), it follows that \( m < 2n \). Therefore, the runtime becomes \( \Theta((2n)^{\lg 7}) = \Theta(2^{\lg 7} \cdot n^{\lg 7}) = \Theta(n^{\lg 7}) \).

Problem B (CLRS 28.2-3)

What is the largest \( k \) such that if you can multiply \( 3 \times 3 \) matrices using \( k \) multiplications, then you can multiply \( n \times n \) matrices in time \( o(n^{\lg 7}) \)? What would the running time of this algorithm be?

Strassen’s algorithm takes the approach of a recursive multiply with a base condition of \( 2 \times 2 \) matrices. We are asked to apply an algorithm using a base condition of a \( 3 \times 3 \) matrix and told it will take \( k \) multiplications.

Consider the comparative recursions:

\[
\text{Strassen:} \quad T(n) = 7T(n/2) + \Theta(n^2) \\
\text{3x3:} \quad T(n) = kT(n/3) + \Theta(n^2)
\]

As the hint points out, case 1 of the Master Theorem applies and the recursive term dominates. Concentrating on the \( 3 \times 3 \) recursion, we want to solve for \( k \) such that the number of multiplies will be less than \( n^{\lg 7} \). We do so as follows:

\[
\Theta(n^{\lg 7}) \geq \Theta(n^{\log_{3}^{3}k}), \text{ so}
\]
$$n^{\lg 7} \geq n^{\log[3]k}$$
$$\lg 7 \geq \log_3 k$$

Utilizing maple as suggested to solve for $k$ we find that $21.8499 \geq k$.
Therefore the largest $k$ possible, while still doing better than $o(n^{\lg 7})$ using this 3x3 method is 21.

Plugging 21 back into the recurrence and solving using case 1 of the Master Theorem provides us with a running time of:

$$\Theta(n^{\log[3](21)}) = \Theta(n^{2.7712})$$