Algorithms

Assignment: Matrix Multiplication

Complexity

- The standard matrix-multiplication algorithm multiplies two $n \times n$ matrices using exactly n^3 multiplications and $n^2(n-1)$ additions.
- Strassen's algorithm multiplies two 2×2 matrices using exactly 7 multiplications and a total of 18 additions and subtractions.
- For $n = 2^k$, where k is a nonnegative integer, the total number of arithmetic operations performed by Strassen's algorithm is

  $$\Theta(n^{\log_2 7}) = \Theta(n^{2.807354922\ldots}) = O(n^{2.81}).$$

  For other values of n, padding both matrices to dimensions equal to the smallest power of two greater than or equal to n preserves this asymptotic bound.

1. Apply Strassen's algorithm recursively, using the 2×2 procedure as the base case. For each of $n = 4, 8, 16$, compute the exact number of multiplications and the exact total number of additions and subtractions required to multiply two $n \times n$ matrices. Include operations performed at every recursion level. Explain how you obtained each count.

   Solution for matrices of size 4×4:

   - 7 multiplications of two 2×2 matrices are required to multiply two 4×4 matrices. Each multiplication of two 2×2 matrices requires 7 multiplications. Altogether,

     $$49 = 7 \cdot 7 = 7^2$$

     multiplications are required to multiply two 4×4 matrices.
   - 7 multiplications of two 2×2 matrices and 18 additions/subtractions of two 2×2 matrices are required to multiply two 4×4 matrices. Each multiplication of two 2×2 matrices requires 18 additions/subtractions and each addition/subtraction of two 2×2 matrices requires 4 additions/subtractions. Altogether,

     $$198 = 7 \cdot 18 + 18 \cdot 4$$

     additions/subtractions are required to multiply two 4×4 matrices.

   Solution for matrices of size 8×8:

   - 7 multiplications of two 4×4 matrices are required to multiply two 8×8 matrices. Each multiplication of two 4×4 matrices requires 7^2 multiplications. Altogether,

     $$343 = 7 \cdot 7^2 = 7^3$$

     multiplications are required to multiply two 8×8 matrices.
   - 7 multiplications of two 4×4 matrices and 18 additions/subtractions of two 4×4 matrices are required to multiply two 8×8 matrices. Each multiplication of two 4×4 matrices requires 198 additions/subtractions and each addition/subtraction of two 4×4 matrices requires 16 additions/subtractions. Altogether,

     $$1674 = 7 \cdot 198 + 18 \cdot 16$$

     additions/subtractions are required to multiply two 8×8 matrices.

   Solution for matrices of size 16×16:

   - 7 multiplications of two 8×8 matrices are required to multiply two 16×16 matrices. Each multiplication of two 8×8 matrices requires 7^3 multiplications. Altogether,

     $$2401 = 7 \cdot 7^3 = 7^4$$

     multiplications are required to multiply two 16×16 matrices.
   - 7 multiplications of two 8×8 matrices and 18 additions/subtractions of two 8×8 matrices are required to multiply two 16×16 matrices. Each multiplication of two 8×8 matrices requires 1674 additions/subtractions and each addition/subtraction of two 8×8 matrices requires 64 additions/subtractions. Altogether,

     $$12870 = 7 \cdot 1674 + 18 \cdot 64$$

     additions/subtractions are required to multiply two 16×16 matrices.

2. Let $M(n)$ be the number of multiplications performed by Strassen's algorithm on two $n \times n$ matrices, where $n = 2^k$ and $k \geq 0$. Establish the recursive formula for $M(n)$, including its base case, and derive an exact closed-form expression for $M(n)$. Ignore additions and subtractions in this problem.

Solution: The recursive call involves no additional multiplications beyond those performed by the 7 multiplications of the smaller matrices. Trivially, only one multiplication is performed when multiplying two 1×1 matrices. As a result, the recursive formula is:

$$\begin{aligned} M(1) &= 1 \\ M(n) &= 7M\left(\frac{n}{2}\right) \text{ for } n \geq 2 \end{aligned}$$

Proposition: The solution to the above recursion is:

$$M(n) = 7^{\log_2(n)} = n^{\log_2(7)}$$

Proof by induction: For $n = 1$, it follows that $M(1) = 1$ and $7^{\log_2(1)} = 7^0 = 1$.
For $n \geq 1$, assume that $M(n) = 7^{\log_2(n)}$ and prove that $M(2n) = 7^{\log_2(2n)}$.

$$\begin{aligned} M(2n) &= 7M(n) \\ &= 7 \cdot 7^{\log_2(n)} \\ &= 7^{\log_2(n)+1} \\ &= 7^{\log_2(2n)} \end{aligned}$$

Top-down proof: Apply the recursion until $n = 1$,

$$\begin{aligned} M(n) &= 7 \cdot M\left(\frac{n}{2}\right) \\ &= 7 \cdot 7 \cdot M\left(\frac{n}{4}\right) \\ &= 7 \cdot 7 \cdot 7 \cdot M\left(\frac{n}{8}\right) \\ &\vdots \\ &= 7^i \cdot M\left(\frac{n}{2^i}\right) \\ &\vdots \\ &= 7^{\log_2(n)} \cdot M\left(\frac{n}{2^{\log_2(n)}}\right) \\ &= 7^{\log_2(n)} \cdot M\left(\frac{n}{n}\right) \\ &= 7^{\log_2(n)} \cdot M(1) \\ &= 7^{\log_2(n)} \cdot 1 \\ &= 7^{\log_2(n)} \end{aligned}$$

Remark: The above line "$= 7^i \cdot M\left(\frac{n}{2^i}\right)$" needs a proof by induction. However, this method helps in finding the solution that allows a direct proof by induction.

3. Let $A(n)$ be the total number of additions and subtractions performed by Strassen's algorithm on two $n \times n$ matrices, where $n = 2^k$ and $k \geq 0$. Establish the recursive formula for $A(n)$, including its base case, and derive an exact closed-form expression for $A(n)$. Ignore multiplications in this problem.

Solution: The seven recursive multiplications contribute $7A(n/2)$ additions/subtractions and $18(n^2/4)$ additions/subtractions due to the 18 additions/subtractions of $(n/2) \times (n/2)$ matrices. Trivially, no additions/subtractions are needed when multiplying two 1×1 matrices. As a result, the recursive formula is:

$$\begin{aligned} A(1) &= 0 \\ A(n) &= 7A(n/2) + 18(n^2/4) \text{ for } n \geq 2 \end{aligned}$$

Proposition: The solution to the above recursion is:

$$A(n) = 6 \cdot 7^{\log_2(n)} - 6 \cdot 4^{\log_2(n)} = 6 \cdot n^{\log_2(7)} - 6 \cdot n^2$$

Proof by induction: For $n = 1$, it follows that $A(1) = 0$ and $6 \cdot 7^{\log_2(1)} - 6 \cdot 1^2 = 0$.
For $n \geq 1$, assume that $A(n) = 6 \cdot 7^{\log_2(n)} - 6 \cdot n^2$ and prove that $A(2n) = 6 \cdot 7^{\log_2(2n)} - 6 \cdot (2n)^2$.

$$\begin{aligned} A(2n) &= 7A(n) + 18 \cdot ((2n)^2/4) \\ &= 7(6 \cdot 7^{\log_2(n)} - 6 \cdot n^2) + 18 \cdot n^2 \\ &= 6 \cdot 7^{\log_2(n)+1} - 42 \cdot n^2 + 18 \cdot n^2 \\ &= 6 \cdot 7^{\log_2(n)+1} - 24 \cdot n^2 \\ &= 6 \cdot 7^{\log_2(2n)} - 6 \cdot (2n)^2 \end{aligned}$$

Top-down proof: Apply the recursion until $n = 1$,

$$\begin{aligned} A(n) &= 7 \cdot A(n/2) + 18(n^2/4) \\ &= 7 \cdot 7 \cdot A(n/4) + 7 \cdot 18(n^2/16) + 18(n^2/4) \\ &= 7 \cdot 7 \cdot 7 \cdot A(n/8) + 7 \cdot 7 \cdot 18(n^2/64) + 7 \cdot 18(n^2/16) + 18(n^2/4) \\ &\vdots \\ &= 7^i \cdot A(n/(2^i)) + 7^{i-1} \cdot 18(n^2/4^i) + \cdots + 7 \cdot 18(n^2/16) + 18(n^2/4) \\ &= 7^i \cdot A(n/(2^i)) + ((7/4)^{i-1} + \cdots + (7/4)^1 + (7/4)^0) \cdot (18/4) \cdot n^2 \\ &= 7^i \cdot A(n/(2^i)) + ((7/4)^i - 1)/(7/4 - 1) \cdot (18/4) \cdot n^2 \\ &= 7^i \cdot A(n/(2^i)) + ((7/4)^i - 1) \cdot (4/3)(18/4) \cdot n^2 \\ &= 7^i \cdot A(n/(2^i)) + ((7/4)^i - 1) \cdot 6 \cdot n^2 \\ &\vdots \\ &= 7^{\log_2(n)} \cdot A(n/(2^{\log_2(n)})) + ((7/4)^{\log_2(n)} - 1) \cdot 6 \cdot n^2 \\ &= 7^{\log_2(n)} \cdot A(1) + (7^{\log_2(n)}/4^{\log_2(n)}) \cdot 6 \cdot n^2 - 6 \cdot n^2 \\ &= 0 + (7^{\log_2(n)}/n^{\log_2(4)}) \cdot 6 \cdot n^2 - 6 \cdot n^2 \\ &= (7^{\log_2(n)}/n^2) \cdot 6 \cdot n^2 - 6 \cdot n^2 \\ &= 6 \cdot 7^{\log_2(n)} - 6 \cdot n^2 \\ &= 6 \cdot n^{\log_2(7)} - 6 \cdot n^2 \end{aligned}$$

Remark: The identity $1 + q + q^2 + \cdots + q^{i-1} = (q^i - 1)/(q - 1)$ was used for $q = 7/4$. Other algebraic justifications for the equalities are omitted.

4. Assume that $n = 3^k$ for some integer $k \geq 1$. Suppose a method for multiplying two 3×3 matrices uses 25 multiplications and a total of 50 additions and subtractions. Consider the recursive algorithm that applies this method to $(n/3) \times (n/3)$ matrix blocks. Let $T(n)$ denote the total number of arithmetic operations performed. Establish the recursive formula for $T(n)$, including its base case, and use the Master Theorem to solve it.

   Solution: Trivially, only one multiplication is needed when multiplying two 1×1 matrices. Therefore, the following is the recursive formula for the complexity of this algorithm:

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 25T(n/3) + 50(n/3)^2 \text{ for } n \geq 3 \end{aligned}$$

   By the Master Theorem, the solution to this recursion is

$$T(n) = \Theta(n^{\log_3 25}) = \Theta(n^{2.929947\ldots}) \ .$$

5. Suppose a method for multiplying two 3×3 matrices uses r multiplications and a constant number of additions and subtractions. Recursively apply this method to $n \times n$ matrices, where n is a power of 3. What is the largest integer value of r for which the resulting algorithm has asymptotically lower arithmetic complexity than Strassen's algorithm? Justify your answer.

   Solution: Denote by r the number of multiplications in multiplying two matrices of size 3×3. Trivially, only one multiplication is needed when multiplying two 1×1 matrices. Therefore, the following is the recursive formula for the complexity of this algorithm:

$$\begin{aligned} T(1) &= 1 \\ T(n) &= rT(n/3) + \Theta(n^2) \text{ for } n \geq 3 \end{aligned}$$

   By the Master Theorem, for $r > 9$, the solution to this recursion is:

$$T(n) = \Theta\left(n^{\log_3(r)}\right)$$

   Since $\log_3(22) > 2.81 > \log_2(7)$ and $\log_3(21) < 2.78 < \log_2(7)$, it follows that in order to get an asymptotically lower arithmetic complexity than Strassen's algorithm r must be at most 21.

6. Assume that $n = 70^k$ for some integer $k \geq 1$. Suppose a fixed method for multiplying two 70×70 matrices uses 143,640 multiplications and a total of α additions and subtractions, where α is a constant independent of n. Consider the recursive algorithm that applies this method to $(n/70) \times (n/70)$ matrix blocks. Let $T(n)$ denote the total number of arithmetic operations performed. Establish the recursive formula for $T(n)$, including its base case, and use the Master Theorem to solve it.

   Solution: Trivially, only one multiplication is needed when multiplying two 1×1 matrices. Therefore, the following is the recursive formula for the complexity of this algorithm:

$$\begin{aligned} T(1) &= 1 \\ T(n) &= 143{,}640T(n/70) + \alpha(n/70)^2 \text{ for } n \geq 70 \end{aligned}$$

   By the Master Theorem, the solution to this recursion is

$$T(n) = \Theta(n^{\log_{70}(143{,}640)}) = \Theta(n^{2.795122689\ldots}) \ .$$