University of Iowa | Homework 8 | CS 3330
Department of CS | Fall 2024

Assigned: Friday 11:59 PM, November 1, 2024
Due: Friday 11:59 PM, November 8, 2024

1. [**A Human Compiler**]: A computer scientist must understand how algorithms work. Compile each of the following algorithms.

   (a) [**10 points**] Execute the "MULTIPLY algorithm" on the given binary numbers, showing the input and the output for each algorithm call. (The order of calls matters! See page 18 of 05DivideAndConquerII for reference.)

	Input	Output
1st Call	$(101, 011, 3)$	1111

   (b) [**10 points**] Execute the "KARATSUBA-MULTIPLY algorithm" on the given binary numbers, showing the input and the output for each algorithm call. (The order of calls matters! See page 21 of 05DivideAndConquerII for reference.)

	Input	Output
1st Call	$(111, 101, 3)$	100011

   (c) [**10 points**] Execute the "STRASSEN algorithm" for the given matrices, showing the input and the output for each algorithm call. (The order of calls matters! See page 32 of 05Divide-AndConquerII for reference.)

	Input	Output
1st Call	$\begin{pmatrix} 1 & 3 \\ 5 & 2 \end{pmatrix}, \begin{pmatrix} -1 & 4 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 7 \\ -5 & 22 \end{pmatrix}$

2. [**Substitution Method**]: The first approach for solving a recurrence is the substitution method, consisting of two steps:

   (a) Begin by guessing the form of the solution, using symbolic constants.
   (b) Use mathematical induction to demonstrate the verify your guess and find the constants.

   Use the substitution method to verify the asymptotic solutions for the following recurrences, or provide a justification for why the suggested solution is incorrect. Note that the Master Theorem does not apply in this case.

   (a) [**5 points**] $T(n) = T(n-1) + n^2$ has solution $T(n) = O(n^3)$.
   (b) [**5 points**] $T(n) = 2T(n-1) + n$ has solution $T(n) = O(n^5)$.

3. [**Recursion Tree**]: The second approach for solving a recurrence is the recursion tree method. Each node represents the cost of a single subproblem. You sum the costs within each level of the tree to

obtain the per-level costs, and then you sum all the per-level costs to determine the total cost of all levels of the recursion. This method is best for building intuition for a good guess.

Using the recursion tree method, guess a good asymptotic upper bound on the solution of the following recurrence.

(a) **[5 points]** $T(n) = 2T(n/5) + n^3$.

4. **[Master Method]**: The third approach for solving a recurrence is the master method, which has three cases that you need to memorize.

   Using the master method, derive tight asymptotic bounds for the following recursions.

   (a) **[5 points]** $T(n) = 5T(n/4) + n^3$

   (b) **[5 points]** $T(n) = 3T(n/8) + n^2$

5. **[Master of Complexity]**: The Master Theorem might seem easy to work with, but in some cases, it may not apply directly. However, a good computer scientist can often find a creative way to make it work. Consider the following recursive relation:

$$T(n) = 3T(n/2) + n^2\sqrt{n}\log\log n$$

   (a) **[5 points]** Can the Master Theorem be applied directly to find the solution of the recursion? Justify your answer.

   (b) **[5 points]** We can express the recursion as $T(n) = 3T(n/2) + \Omega(n^\alpha)$. What is the largest value of α? Justify your answer and use the master theorem to solve this recursion.

   (c) **[5 points]** We can also rewrite the recursion as $T(n) = 3T(n/2) + O(n^\beta)$. What are the possible choices for β? Justify your answer and use the master theorem to solve this recursion.

   (d) **[5 points]** Find a function $f(n)$ such that $T(n) = \theta(f(n))$ is the solution of the recursion.

6. **[Rival Research Groups]**: Two rival teams of mathematicians, Team X and Team Y, are working on similar problems. Unlike "most researchers", their primary objective is to maximize their academic publications! To stay ahead, Team X has been carefully observing the techniques and methods that Team Y is developing. Assist Team X in extending the techniques that Team Y has recently developed.

   (a) **[15 points]** Following the divide-and-conquer paradigm, Team Y has developed an algorithm that takes two $m \times m$ matrices as input and returns their product using k scalar multiplications. Based on this algorithm, develop an $n \times n$ matrix multiplication algorithm, assuming $n = m^l$. Write the recurrence relation for the running time complexity of this algorithm. What is the time complexity of your algorithm?

   (b) **[5 points]**: Assuming $m = 48$, what is the largest value of k for which the resulting algorithm has a complexity of $O(n^{2.79})$? Justify your answer.

   (c) **[5 points]**: Team Y has developed an algorithm for matrix squaring with a time complexity of $O(n^{2.3})$. Extend their results to develop an algorithm for matrix multiplication with the same time complexity of $O(n^{2.3})$.