Advanced Data Structures and Algorithms

Comprehensive Assignment Solutions

Arya Raag
Roll No: A125002

M.Tech (Computer Science and Engineering)

January 5, 2026

1 Matrix Decompositions

Question 2. Solve the following recurrence relation arising from the LUP decomposition solve procedure:

$$T(n) = \sum_{i=1}^{n} \left[O(1) + \sum_{j=1}^{i-1} O(1) \right] + \sum_{i=1}^{n} \left[O(1) + \sum_{j=i+1}^{n} O(1) \right]$$

Detailed Solution:

Context of the Recurrence

This recurrence models the computational cost of the two-step solution process for linear systems after decomposition:

1. **Forward Substitution ($Ly = Pb$):** Represented by the first summation term.
2. **Backward Substitution ($Ux = y$):** Represented by the second summation term.

Analysis of the First Term (Forward Substitution)

Let T_1 be the first summation:

$$T_1 = \sum_{i=1}^{n} \left[c_1 + \sum_{j=1}^{i-1} c_2 \right]$$

where c_1, c_2 are constants ($O(1)$). The inner sum runs $(i-1)$ times. Thus:

$$T_1 \approx \sum_{i=1}^{n} (c_1 + c_2(i-1))$$

This is an arithmetic progression involving i.

$$T_1 = c_1 n + c_2 \sum_{i=1}^{n} (i-1)$$

Using the summation formula $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$:

$$T_1 \approx c_2 \frac{n^2}{2} = O(n^2)$$

Analysis of the Second Term (Backward Substitution)

Let T_2 be the second summation:

$$T_2 = \sum_{i=1}^{n} \left[c_3 + \sum_{j=i+1}^{n} c_4 \right]$$

The inner sum runs from $j = i+1$ to n, which is exactly $n-(i+1)+1 = n-i$ iterations.

$$T_2 \approx \sum_{i=1}^{n} (c_3 + c_4(n-i))$$

Let $k = n-i$. As i goes from 1 to n, k goes from $n-1$ to 0.

$$T_2 \approx \sum_{k=0}^{n-1} (c_3 + c_4 k)$$

This is also an arithmetic progression summing to $O(n^2)$.

Total Complexity

Adding both components:

$$T(n) = T_1 + T_2 = O(n^2) + O(n^2) = O(n^2)$$

Conclusion

We have analytically derived that solving a linear system $Ax = b$ given the LUP decomposition takes $O(n^2)$ time. This highlights the efficiency of the decomposition strategy: if we need to solve for multiple vectors b, we only perform the expensive $O(n^3)$ decomposition once, and each subsequent solve is a fast $O(n^2)$ operation.
