Week 13 Summary: Interval DP, O(n log n) LIS, and Bitmask TSP

Z5007 Programming and Data Structures · MTech DSAI · Odd Semester 2026-27 · Dr. Innocent Nyalala

1. Interval DP: Matrix Chain Multiplication

Multiplying matrices $A_1 \cdots A_n$ with dimensions $p_0 \times p_1, \ldots, p_{n-1} \times p_n$ varies drastically by parenthesization. For every contiguous subchain $[i, j]$, test all split indices $k \in [i, j-1]$ in increasing length order:

$$m[i][j] = \min_{i \le k < j} \big(m[i][k] + m[k+1][j] + p_i p_{k+1} p_{j+1} \big), \qquad O(n^3) \text{ time, } O(n^2) \text{ space.}$$

2. Longest Increasing Subsequence in $O(n \log n)$

Maintain array `tails` where `tails[L]` stores the smallest ending value of any increasing subsequence of length $L+1$. Because `tails` stays strictly sorted, each new element updates one entry via binary search (`bisect_left`) in $O(\log n)$.

3. Bitmask Dynamic Programming (Held-Karp TSP)

State dp[mask][u] stores the shortest path visiting the vertex subset encoded by mask and terminating at u. Transitioning over unvisited neighbors $v \notin$ mask reduces naive $O(n!)$ search to $O(n^2 2^n)$.

Worked Check

Dimensions $[30, 35, 15, 5, 10, 20, 25]$ achieve optimal scalar product count $15,125$. On $[10, 9, 2, 5, 3, 7, 101, 18]$, patience `tails` grows to length 4.