Welcome to CS473 (CS 498 DL1) Algorithms

Jeff
Dynamic Programming

Optimization/Construction problem

→ Series of decisions

↑ what decisions?

Input: [Diagram showing input structure]

Output: [Diagram showing output structure]

Formulate decisions as a recursive problem

→ specify

→ recurrence

Correct

Make the recurrence iterative

Efficient
Longest Increasing Subsequence

1 8 7 8 7 8 4 1 1 4 8 9 4 9 1 1 8 9 8 9 8

1. Seq. of decisions

Is this in the LIS assuming prev decisions are correct?

2. Problem: \( \text{LIS}(i, j) = \text{Length of longest increasing subsequence of } A[j \ldots n] \) such that \( \text{all entries are larger than } A[i] \)

\( \text{LIS}(A) = \& \text{LIS}(0, 1) \)
3. Recurrence:

\[
LIS(i,j) = \begin{cases} 
0 & \text{if } j > n \\
LIS(i, j+1) & \text{if } A(j) \leq A[i] \\
\max \{LIS(i, j+1), 1+LIS(j+1)\} & \text{otherwise}
\end{cases}
\]
Iterative

Memoization structure:

array $\text{LIS}[0..n, 1..n+1]$

Answer!

Order:
for $j \leftarrow n$ down to 1
for $i \leftarrow 0$ to $j - 1$

$O(n^2)$ time

Recurrence
Decisions: What's next in the output subseq?

Problem: \( \text{LIS}(i) = \)
Length of longest increasing subseq. of \( A[1..n] \) that starts with \( A[i] \)

Global: \( A[0] = -\infty \)
return \( \text{LIS}(0) - 1 \)

\( \text{LIS}(i) = \begin{cases} 
1 + \max \left\{ \text{LIS}(j) \mid j < i \leq n, A[j] > A[i] \right\} 
\end{cases} \)
LIS \{1...n\} 

O(n^2) time

For \( i = n \) to 0

reccurrence