CompSci 161
Winter 2023 Lecture 13:
Dynamic Programming IV:
Longest Increasing Subsequence

Problem Statement

Example:

30	10	20	40	60	50	70	90	80

First Algorithm

LIS(0,1) - 1

Think: A[i] most recent added. Add A[j]?

- $A[0] = -\infty$ (for simplicity)
- LISBigger(i, j) : the LIS of $A[j \ldots n]$ w $> A[i]$ with
- What is top-level call?

O(n^2) cases

O(1) time each

if j>n return 0

skip = LISBigger(i, j+1)

if A[i] $\geq$ A[j] return skip

use = LISBigger(j, j+1)+1

return max(skip, use)

Iterative Version

Decl LISB[0...n, 1..n+1]

$A[0] = -\infty$

for $i = 0 \dots n$
  LISB[i, n+1] = 0

- skip $=$ LISB$(i, j+1)$
- use $=$ LISB$(j, j+1) + 1$

for j = n ... 1 // (decreasing)
  for i = 0 ... j-1
    skip = LISB[i, j+1]
    if A[i] ≥ A[j]   LISB[i, j] = skip
    else
      use = LISB[j, j+1] + 1
      LISB[i, j] = max(use, skip)

Improved Algorithm

LISFirst(i) : LIS w/ i as 1^{st}

$$\max \{ \underbrace{1 + \text{LISFirst}(j)}_{\text{evaluate this}} : \underbrace{j > i \text{ and } A[j] > A[i]}_{\text{for all where this is true.}} \}$$

Good review: do iterative :)

Iterative Version

$$\text{LISFirst} = 1 + \max\{\text{LISFirst}(j) : j > i \text{ and } A[j] > A[i]\}$$

Demonstration

0
8
4
12
2
10
6
14
1
9
5
13
3
11
7
15