# CPSC 420 Lecture 13: Today's announcements:

- HW2 is on Gradescope, due Feb 9, 23:59
- Examlet 2 on Feb 17 in class. Closed book & no notes
- Reading: Ch.H Linear Programming, Ch.3 Dynamic Programming [all by Erickson]

## Today's Plan

- Linear programming duality
- Dynamic programming
  - Longest Increasing Subsequence
  - Edit Distance

# Longest Increasing Subsequence

What is a longest increasing subsequence of:

5 3 4 9 6 2 1 8

A sequence $S[1..k]$ is **increasing** if $S[i] < S[i+1]$ $\forall i = 1..k-1$.

Given: A sequence of numbers $R[1..n]$. Find: LIS of $R$

Use LCS to solve LIS

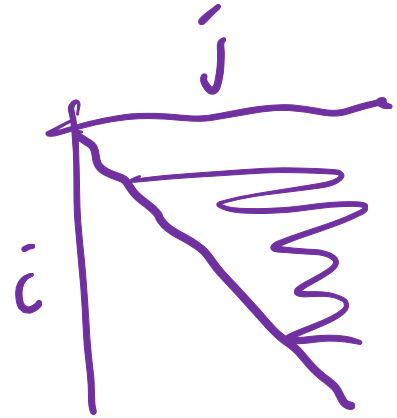
LIS($R$)
1. $S =$ SORT($R$) remove duplicates
2. output LCS($S, R$)

Running time? $O(n^2)$

# Recursive Longest Increasing Subsequence

Idea: Either take $R[n]$ or don't.

LimitedLIS($R$, $x$)
# Find LIS with last number less than $x$.
1. if $|R| = 0$ return []
2. $T = []$
3. if $R[n] < x$ then
4. $\quad T = \text{LimitedLIS}(R[1..n-1], R[n]) \circ R[n]$
5. return $\max\{T, \text{LimitedLIS}(R[1..n-1], x)\}$




Running time? $T(n) \approx 2T(n-1)$ $\Omega(n^2)$

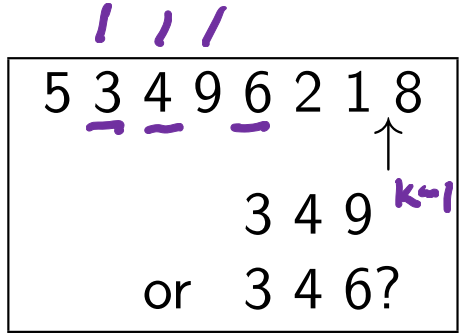
Calls to LLIS $(R[1...i], R[j])$ $j > i$

$\Theta(n^2)$ even with Dyn. Prog.

# Faster Longest Increasing Subsequence

To find LIS($R[1..k]$), what information about $R[1..k-1]$ is enough?

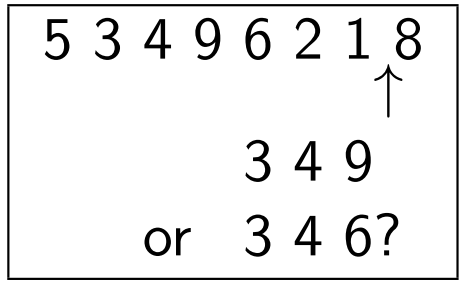
A. LIS of $R[1..k-1]$



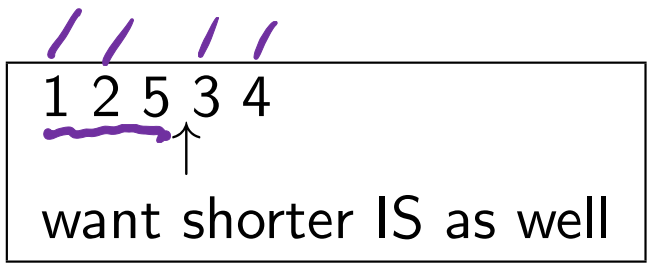
# Faster Longest Increasing Subsequence

To find LIS($R[1..k]$), what information about $R[1..k-1]$ is enough?

A. LIS of $R[1..k-1]$



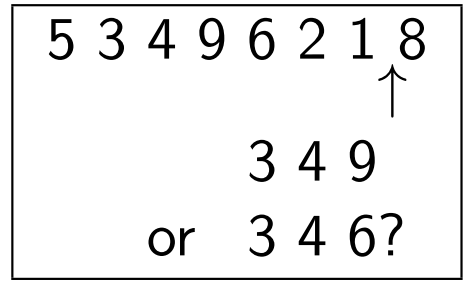

B. Best LIS of $R[1..k-1]$



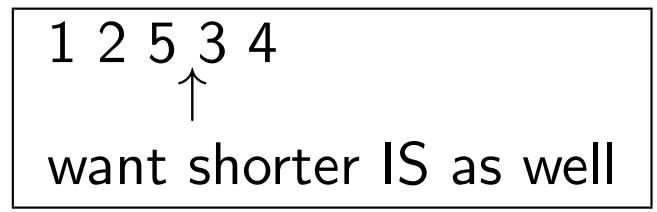
# Faster Longest Increasing Subsequence

To find LIS($R[1..k]$), what information about $R[1..k-1]$ is enough?

A. LIS of $R[1..k-1]$




B. Best LIS of $R[1..k-1]$

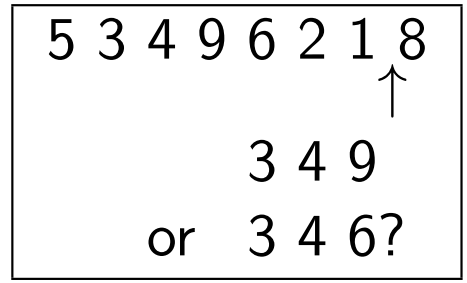



C. Best ISs of length $1, 2, \ldots, j$, where $j = |\text{LIS}(R[1..k-1])|$

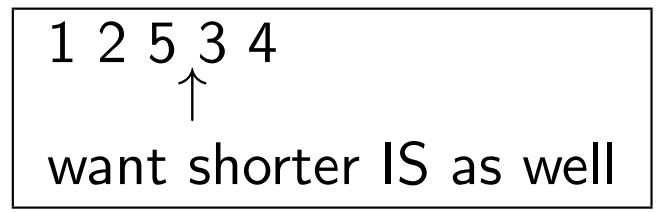
# Faster Longest Increasing Subsequence

To find LIS($R[1..k]$), what information about $R[1..k-1]$ is enough?

A. LIS of $R[1..k-1]$




B. Best LIS of $R[1..k-1]$



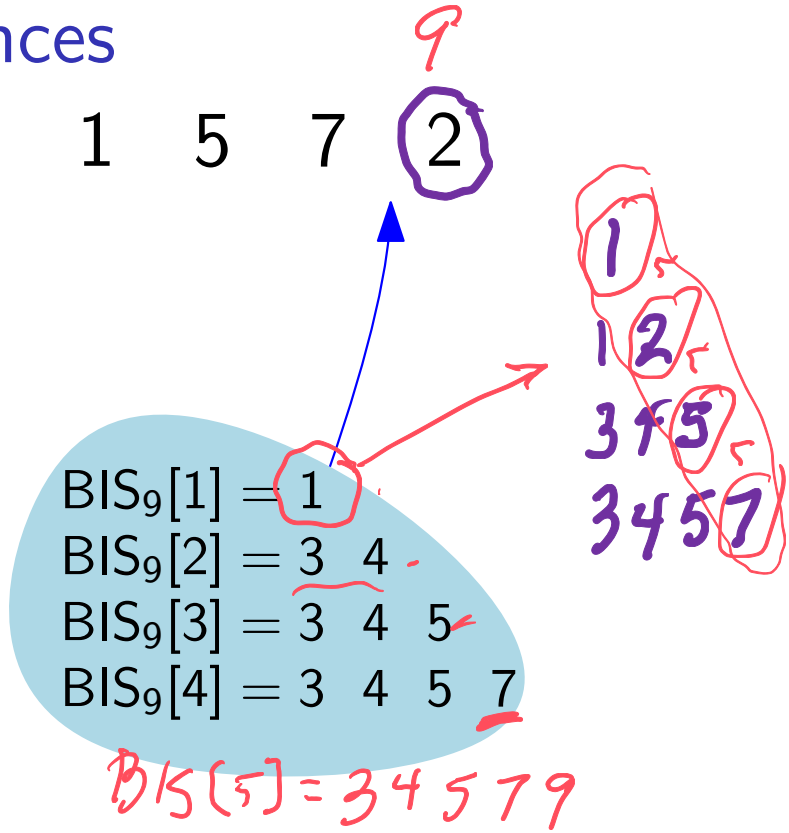

C. Best ISs of length $1, 2, \ldots, j$, where $j = |\text{LIS}(R[1..k-1])|$

Now we need to find best ISs for $R[1..k]$ using this info. How?

# Recursive Best Increasing Subsequences

8 3 4 9 6 2 1 5 7 2

$R[k]$ **extends** $\mathrm{BIS}[j]$ iff $\mathrm{BIS}[j].\mathit{last} < R[k]$ and
$(j = |\mathrm{BIS}|$ or $\mathrm{BIS}[j+1].\mathit{last} > R[k])$




Claim: $\mathrm{BIS}[1].\mathit{last} < \mathrm{BIS}[2].\mathit{last} < \cdots$

Proof: If $\mathrm{BIS}[i].\mathit{last} \geq \mathrm{BIS}[i+1].\mathit{last}$ then the first $i$ numbers in $\mathrm{BIS}[i+1]$ would be a better $\mathrm{BIS}[i]$ $\Rightarrow\Leftarrow$.

# Recursive Best Increasing Subsequences

8 3 4 9 6 2 1 5 7 2

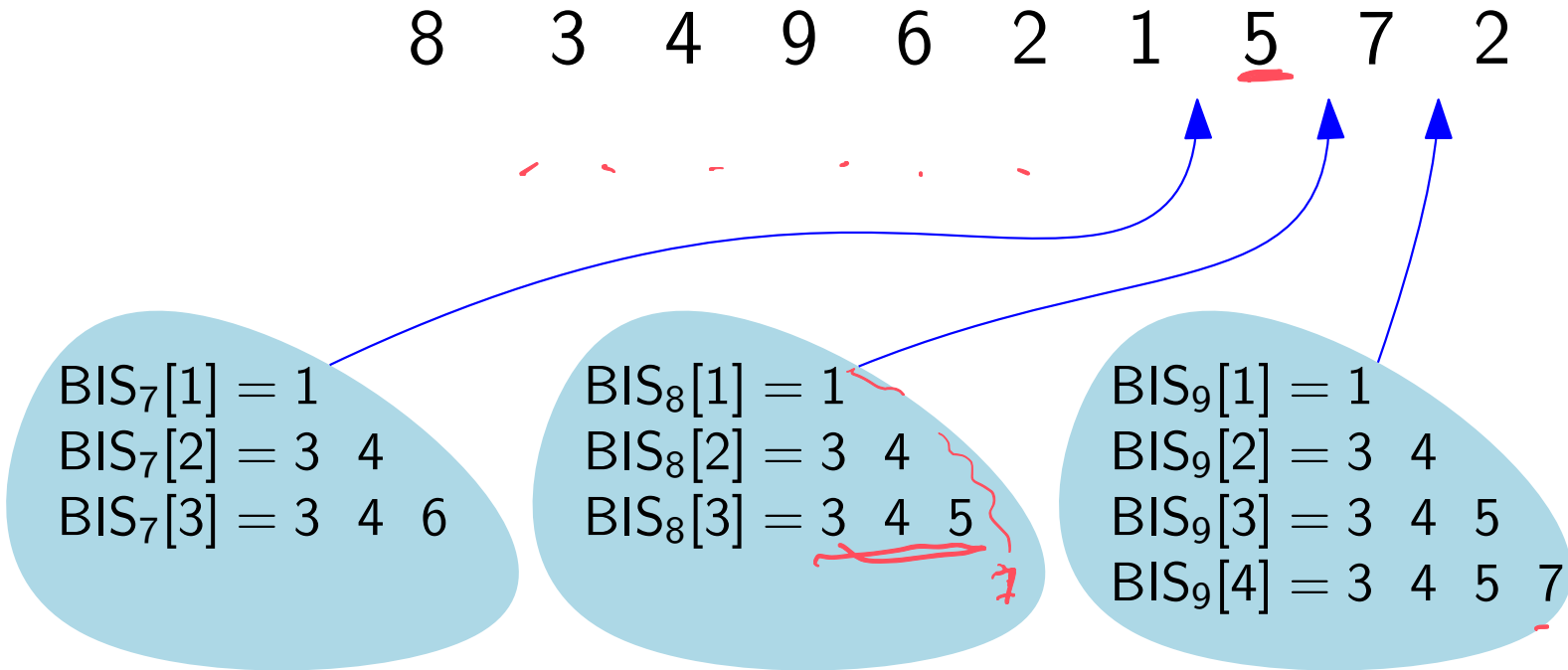
$R[k]$ **extends** $\text{BIS}[j]$ iff $\text{BIS}[j].\mathit{last} < R[k]$ and
$(j = |\text{BIS}| \text{ or } \text{BIS}[j+1].\mathit{last} > R[k])$

$\text{BIS}_9[1] = 1$
$\text{BIS}_9[2] = 3\ \ 4$
$\text{BIS}_9[3] = 3\ \ 4\ \ 5$
$\text{BIS}_9[4] = 3\ \ 4\ \ 5\ \ 7$

$\underline{\text{BIS}(R)}$
1. if $|R| = 0$ return []
2. if $|R| = 1$ return $[R[1]]$
3. $B = \text{BIS}(R[1..n-1])$ ← recurse
4. for $j = 1$ to $|B| - 1$
5. if $R[n]$ extends $B[j]$ then
$B[j+1] = B[j] \circ R[n]$; return $B$
6. return $B$

replace with binary search

# Best Increasing Subsequences Dynamic Programming




$R[k]$ **extends** $\mathrm{BIS}[j]$ iff $\mathrm{BIS}[j].\mathit{last} < R[k]$ and

$$(j = |\mathrm{BIS}| \text{ or } \mathrm{BIS}[j+1].\mathit{last} > R[k])$$

# Best Increasing Subsequences Dynamic Programming

$$8 \quad 3 \quad 4 \quad 9 \quad 6 \quad 2 \quad 1 \quad 5 \quad 7 \quad 2$$

$R[k]$ **extends** $\mathrm{BIS}[j]$ iff $\mathrm{BIS}[j].\mathit{last} < R[k]$ and
$(j = |\mathrm{BIS}| \text{ or } \mathrm{BIS}[j+1].\mathit{last} > R[k])$

Maintain lists $L[0], L[1], ..$ where $L_0[0] = [0]$. After we scan $R[1..k]$

$$L_k[j] = \begin{cases} L_{k-1}[j] \circ R[k] & \text{if } R[k] \text{ extends } \mathrm{BIS}_{k-1}[j] \\ L_{k-1}[j] & \text{otherwise} \end{cases}$$

$L[0] = 0$
$L[1] =$
$L[2] =$
$L[3] =$
$L[4] =$

As a result list $L_k[i]$ ends with $\mathrm{BIS}_k[j].\mathit{last}$ at step $k$.

# Best Increasing Subsequences Dynamic Programming

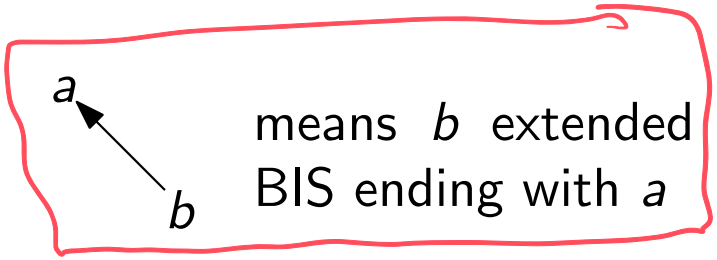
8 3 4 9 6 2 1 5 7 2

$R[k]$ **extends** $\text{BIS}[j]$ iff $\text{BIS}[j].\mathit{last} < R[k]$ and
$(j = |\text{BIS}| \text{ or } \text{BIS}[j+1].\mathit{last} > R[k])$

Maintain lists $L[0], L[1], ..$ where $L_0[0] = [0]$. After we scan $R[1..k]$

$$L_k[j] = \begin{cases} L_{k-1}[j] \circ R[k] & \text{if } R[k] \text{ extends } \text{BIS}_{k-1}[j] \\ L_{k-1}[j] & \text{otherwise} \end{cases}$$

$L[0] = 0$
$L[1] = 8\ 3\ 2\ 1$
$L[2] = 4$
$L[3] = 9\ 6\ 5$
$L[4] = 7\ 6$




use binary search to find $j$

Running time?

As a result list $L_k[i]$ ends with $\text{BIS}_k[j].\mathit{last}$ at step $k$.

# Use LIS to solve LCS [Hunt & Szymanski 1977]

| | A | B | C | B | A | C | C | B |
|---|---|---|---|---|---|---|---|---|
| B | | x | | x | | | | x |
| C | | | x | | | x | x | |
| D | | | | | | | | |
| A | x | | | | x | | | |
| B | | x | | x | | | | x |
| C | | | x | | | x | x | |
| C | | | x | | | x | x | |

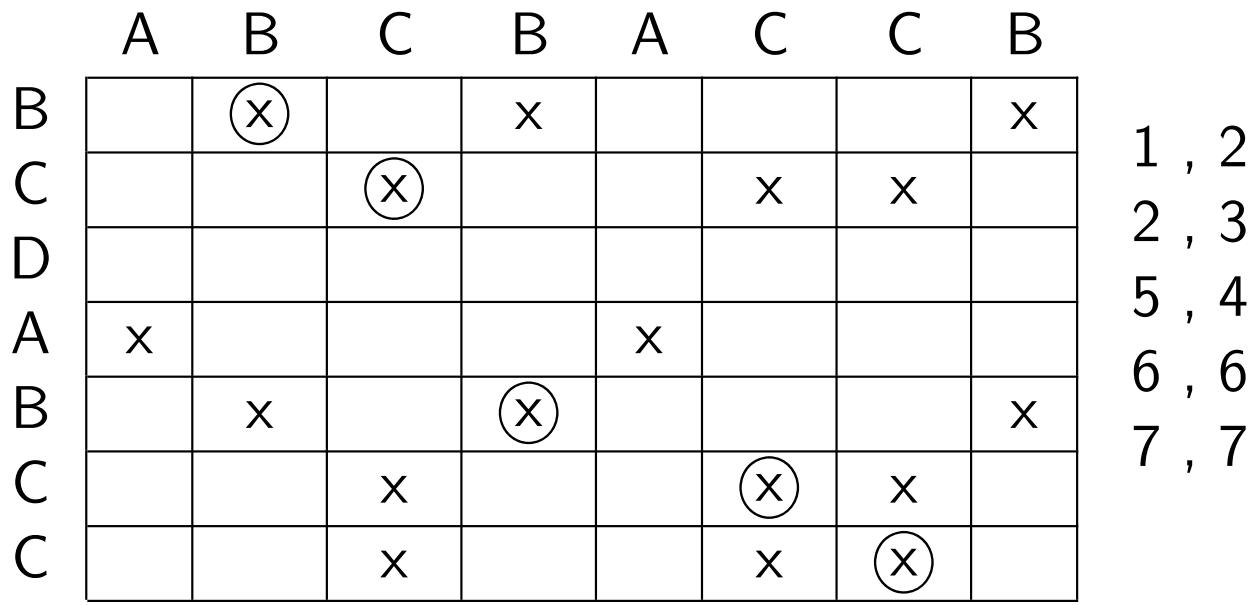
# Use LIS to solve LCS [Hunt & Szymanski 1977]

| | A | B | C | B | A | C | C | B | |
|---|---|---|---|---|---|---|---|---|---|
| B | | (x) | | x | | | | x | 1 , 2 |
| C | | | (x) | | | x | x | | 2 , 3 |
| D | | | | | | | | | 5 , 4 |
| A | x | | | | x | | | | 6 , 6 |
| B | | x | | (x) | | | | x | 7 , 7 |
| C | | | x | | | (x) | x | | |
| C | | | x | | | x | (x) | | |

Longest Common Subsequence = BCBCC

What properties do the circled entries have?

# Use LIS to solve LCS [Hunt & Szymanski 1977]




Longest Common Subsequence = BCBCC

What properties do the circled entries have?

They form a sequence of index pairs of matches that increases in both dimensions.