Dynamic Programming

Four-step (two-phase) method:

1. Characterize the structure of an optimal solution
2. Recursively define the value of an optimal solution
3. Compute the value of an optimal solution in a bottom-up fashion
4. Construct an optimal solution from computed information

Longest Common Subsequence (LCS) – DP case study 3

Problem statement:

Longest Common Subsequence (LCS) – DP case study 3

Problem statement:

Input: *Sequences*

$$\begin{aligned} X_m &= \langle x_1, x_2, x_3, \ldots, x_m \rangle \\ Y_n &= \langle y_1, y_2, \ldots, y_n \rangle \end{aligned}$$

Longest Common Subsequence (LCS) – DP case study 3

Problem statement:

Input: *Sequences*

$$X_m = \langle x_1, x_2, x_3, \ldots, x_m \rangle$$
$$Y_n = \langle y_1, y_2, \ldots, y_n \rangle$$

Output: *longest common subsequence (LCS)* *of* X_m *and* Y_n

LCS

Terminology

1. Sequence, e.g.
   - $X_7 = \langle A, B, C, B, D, A, B \rangle$
   - ALGORITHM

LCS

Terminology

1. Sequence, e.g.
   - $X_7 = \langle A, B, C, B, D, A, B \rangle$
   - ALGORITHM
2. Subsequence, e.g.
   - $\langle A, C, D, B \rangle$ is a subsequence of X
   - ART is a subsequence ALGORITHM

LCS

Terminology

1. Sequence, e.g.
   - $X_7 = \langle A, B, C, B, D, A, B \rangle$
   - ALGORITHM
2. Subsequence, e.g.
   - $\langle A, C, D, B \rangle$ is a subsequence of X
   - ART is a subsequence ALGORITHM
3. Common subsequence, e.g.
   - Given $X_7 = \langle A, B, C, B, D, A, B \rangle$
     $Y_6 = \langle B, D, C, A, B, A \rangle$
   - $Z_3 = \langle B, C, A \rangle$ is a common subsequence of X_7 and Y_6
   - $Z_4 = \langle B, C, B, A \rangle$ is also a common subsequence of X_7 and Y_6

LCS

Terminology

1. Sequence, e.g.
   - $X_7 = \langle A, B, C, B, D, A, B \rangle$
   - ALGORITHM

2. Subsequence, e.g.
   - $\langle A, C, D, B \rangle$ is a subsequence of X
   - ART is a subsequence ALGORITHM

3. Common subsequence, e.g.
   - Given $X_7 = \langle A, B, C, B, D, A, B \rangle$
     $Y_6 = \langle B, D, C, A, B, A \rangle$
   - $Z_3 = \langle B, C, A \rangle$ is a common subsequence of X_7 and Y_6
   - $Z_4 = \langle B, C, B, A \rangle$ is also a common subsequence of X_7 and Y_6

4. Longest common subsequence (LCS), e.g.
   - Z_4 is a longest common subsequence (LCS) of X_7 and Y_6
   - LCS is not unique, $\langle B, C, A, B \rangle$ is also a LCS.

LCS

A brute-force solution:

- For every subsequence of X_m, check if it is a subsequence of Y_n.

LCS

A brute-force solution:

- For every subsequence of X_m, check if it is a subsequence of Y_n.
- Running time: $\Theta(n \cdot 2^m)$
- Intractable!

LCS

DP – step 1: *characterize the structure of an optimal solution*

LCS

DP – step 1: *characterize the structure of an optimal solution*

Let $Z_k = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of

$$X_m = \langle x_1, x_2, \ldots, x_m \rangle \quad \text{and} \quad Y_n = \langle y_1, \ldots, y_n \rangle$$

Then

LCS

DP – step 1: *characterize the structure of an optimal solution*

Let $Z_k = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of

$$X_m = \langle x_1, x_2, \ldots, x_m \rangle \quad \text{and} \quad Y_n = \langle y_1, \ldots, y_n \rangle$$

Then

- Case 1. If $x_m = y_n$, then

  (a) $z_k = x_m = y_n$

  (b) $Z_{k-1} = \langle z_1, z_2, \ldots, z_{k-1} \rangle = \text{LCS}(X_{m-1}, Y_{n-1})$

LCS

DP – step 1: *characterize the structure of an optimal solution*

Let $Z_k = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of

$$X_m = \langle x_1, x_2, \ldots, x_m \rangle \quad \text{and} \quad Y_n = \langle y_1, \ldots, y_n \rangle$$

Then

- Case 1. If $x_m = y_n$, then

  (a) $z_k = x_m = y_n$

  (b) $Z_{k-1} = \langle z_1, z_2, \ldots, z_{k-1} \rangle = \text{LCS}(X_{m-1}, Y_{n-1})$

- Case 2. If $x_m \neq y_n$, then

  (a) $z_k \neq x_m \Longrightarrow Z_k = \text{LCS}(X_{m-1}, Y_n)$

  (b) $z_k \neq y_n \Longrightarrow Z_k = \text{LCS}(X_m, Y_{n-1})$

LCS

DP – step 1: *characterize the structure of an optimal solution*

Let $Z_k = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of

$$X_m = \langle x_1, x_2, \ldots, x_m \rangle \quad \text{and} \quad Y_n = \langle y_1, \ldots, y_n \rangle$$

Then

- Case 1. If $x_m = y_n$, then

  (a) $z_k = x_m = y_n$

  (b) $Z_{k-1} = \langle z_1, z_2, \ldots, z_{k-1} \rangle = \text{LCS}(X_{m-1}, Y_{n-1})$

- Case 2. If $x_m \neq y_n$, then

  (a) $z_k \neq x_m \implies Z_k = \text{LCS}(X_{m-1}, Y_n)$

  (b) $z_k \neq y_n \implies Z_k = \text{LCS}(X_m, Y_{n-1})$

In words, the optimal solution to the (whole) problem contains within it the otpimal solutions to subproblems

LCS

DP – step 1: *characterize the structure of an optimal solution*

Let $Z_k = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of

$$X_m = \langle x_1, x_2, \ldots, x_m \rangle \quad \text{and} \quad Y_n = \langle y_1, \ldots, y_n \rangle$$

Then

- Case 1. If $x_m = y_n$, then

  (a) $z_k = x_m = y_n$

  (b) $Z_{k-1} = \langle z_1, z_2, \ldots, z_{k-1} \rangle = \text{LCS}(X_{m-1}, Y_{n-1})$

- Case 2. If $x_m \neq y_n$, then

  (a) $z_k \neq x_m \Longrightarrow Z_k = \text{LCS}(X_{m-1}, Y_n)$

  (b) $z_k \neq y_n \Longrightarrow Z_k = \text{LCS}(X_m, Y_{n-1})$

In words, the optimal solution to the (whole) problem contains within it the otpimal solutions to subproblems = **the optimal substructure property**

LCS

DP – step 2: *recursively define the value of an optimal solution*

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define

$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define
$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$
- $c[m,n] = \text{length of LCS}(X_m, Y_n)$

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define
$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$
- $c[m,n]$ = length of LCS(X_m, Y_n)
- $c[i,0] = c[0,j] = 0$ for initialization

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define

$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$

- $c[m,n] =$ length of LCS(X_m, Y_n)
- $c[i,0] = c[0,j] = 0$ for initialization
- By Case 1 of the optimal structure property: if $x_i = y_j$, then

  (a) $z_\ell = x_i = y_j$
  (b) $Z_{\ell-1} = \langle z_1, z_2, \ldots, z_{\ell-1} \rangle =$ *LCS*(X_{i-1}, Y_{j-1})

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define
$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$
- $c[m,n] = \text{length of LCS}(X_m, Y_n)$
- $c[i,0] = c[0,j] = 0$ for initialization
- By Case 1 of the optimal structure property: if $x_i = y_j$, then

  (a) $z_\ell = x_i = y_j$
  (b) $Z_{\ell-1} = \langle z_1, z_2, \ldots, z_{\ell-1} \rangle = \mathit{LCS}(X_{i-1}, Y_{j-1})$

  we have
$$c[i,j] = c[i-1, j-1] + 1$$

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define
$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$
- $c[m,n] = \text{length of LCS}(X_m, Y_n)$
- $c[i,0] = c[0,j] = 0$ for initialization
- By Case 1 of the optimal structure property: if $x_i = y_j$, then

  (a) $z_\ell = x_i = y_j$
  (b) $Z_{\ell-1} = \langle z_1, z_2, \ldots, z_{\ell-1} \rangle = LCS(X_{i-1}, Y_{j-1})$

  we have
$$c[i,j] = c[i-1, j-1] + 1$$
- By Case 2 of the optimal structure property: if $x_i \neq y_j$, then

  (a) $z_\ell \neq x_i \Longrightarrow Z_\ell = LCS(X_{i-1}, Y_j)$
  (b) $z_\ell \neq y_j \Longrightarrow Z_\ell = LCS(X_i, Y_{j-1})$

LCS

DP – step 2: *recursively define the value of an optimal solution*

- Define
$$c[i,j] = \text{length of LCS}(X_i, Y_j)$$
- $c[m,n] =$ length of $\text{LCS}(X_m, Y_n)$
- $c[i,0] = c[0,j] = 0$ for initialization
- By Case 1 of the optimal structure property: if $x_i = y_j$, then

  (a) $z_\ell = x_i = y_j$
  (b) $Z_{\ell-1} = \langle z_1, z_2, \ldots, z_{\ell-1} \rangle = LCS(X_{i-1}, Y_{j-1})$

  we have
$$c[i,j] = c[i-1, j-1] + 1$$
- By Case 2 of the optimal structure property: if $x_i \neq y_j$, then

  (a) $z_\ell \neq x_i \Longrightarrow Z_\ell = LCS(X_{i-1}, Y_j)$
  (b) $z_\ell \neq y_j \Longrightarrow Z_\ell = LCS(X_i, Y_{j-1})$

  we have
$$c[i,j] = \max\{c[i,j-1], c[i-1,j]\}$$

LCS

- In summary,

$$c[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \text{ (initials)} \\ c[i-1, j-1] + 1 & \text{if } x[i] = y[j] \quad \text{(Case 1)} \\ \max\{c[i, j-1], c[i-1, j]\} & \text{if } x[i] \neq y[j] \quad \text{(Case 2)} \end{cases}$$

LCS

- In summary,

$$c[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \text{ (initials)} \\ c[i-1,j-1]+1 & \text{if } x[i] = y[j] \quad \text{(Case 1)} \\ \max\{c[i,j-1], c[i-1,j]\} & \text{if } x[i] \neq y[j] \quad \text{(Case 2)} \end{cases}$$

- Meanwhile, create $b[i,j]$ to record the optimal subproblem solution chosen when computing $c[i,j]$

LCS

DP – step 3: *compute $c[i,j]$ (and $b[i,j]$) in a bottom-up approach*

- Compute $c[i,j]$ and $b[i,j]$ in a bottom-up approach.
  - $c[i,j]$ is the length of LCS(X_i, Y_j)
  - $b[i,j]$ shows how to construct the corresponding LCS(X_i, Y_j)

LCS

DP – step 3: *compute $c[i,j]$ (and $b[i,j]$) in a bottom-up approach*

- Compute $c[i,j]$ and $b[i,j]$ in a bottom-up approach.
  - $c[i,j]$ is the length of LCS(X_i, Y_j)
  - $b[i,j]$ shows how to construct the corresponding LCS(X_i, Y_j)
- Cost:
  - Running time: $\Theta(mn)$
  - Space: $\Theta(mn)$

LCS

```
LCS-length(X,Y)
set c[i,0] = 0 and c[0,j] = 0
for i = 1 to m // Row-major order to compute c and b arrays
    for j = 1 to n
        if X(i) = Y(j)
           c[i,j] = c[i-1,j-1] + 1
           b[i,j] = 'Diag'      // go to up diagonal
        elseif c[i-1,j] >= c[i,j-1]
           c[i,j] = c[i-1,j]
           b[i,j] = 'Up'      // go up
        else
           c[i,j] = c[i,j-1]
           b[i,j] = 'Left'      // go left
        endif
    endfor
endfor
return c and b
```

LCS

DP – step 4: *construct an optimal solution from computed information*

LCS

Example: $X_7 = \langle A, B, C, B, D, A, B \rangle$ and $Y_6 = \langle B, D, C, A, B, A \rangle$

LCS

Example: $X_7 = \langle A, B, C, B, D, A, B \rangle$ and $Y_6 = \langle B, D, C, A, B, A \rangle$

$c[\cdot,\cdot] + b[\cdot,\cdot]$:

i \ j		0	1	2	3	4	5	6
		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B	0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D	0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A	0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B	0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

LCS

Example: $X_7 = \langle A, B, C, B, D, A, B \rangle$ and $Y_6 = \langle B, D, C, A, B, A \rangle$

$c[\cdot,\cdot] + b[\cdot,\cdot]$:

i \ j		0	1	2	3	4	5	6
		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B	0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D	0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A	0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B	0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

(1) Length of LCS = c[7,6] = 4
(2) By the b-table ("↑, ←, ↖"), the LCS is $B\,C\,B\,A$