# طراحی و تحلیل الگوریتم ها

**دکتر امیر لکی زاده**

**استادیار گروه مهندسی کامپیوتر دانشگاه قم**

# ۳. Longest Common Subsequence(LCS)

معیارهای شباهت دو رشته :

- یک رشته، شامل رشته دیگری باشد.
- با تغییرات اندکی در رشته اول، رشته دوم به دست می آید.
- یک رشته سومی(دنباله مشترک) با ترتیب یکسان در هر دو رشته ورودی وجود دارد.

$X = \langle x_1, x_2, \ldots, x_m \rangle$

$Z = \langle z_1, z_2, \ldots, z_k \rangle$ is a ***subsequence*** of $X$ if there exists a strictly increasing sequence $\langle i_1, i_2, \ldots, i_k \rangle$ of indices of $X$ such that for all $i = 1, 2, \ldots, k$, we have $x_{i_j} = z_j$. For example, $Z = \langle B, C, D, B \rangle$ is a subsequence of $X = \langle A, B, C, B, D, A, B \rangle$ with corresponding index sequence $\langle 2, 3, 5, 7 \rangle$.

# ۳. Longest Common Subsequence(LCS)

$X = \langle A, B, C, B, D, A, B \rangle$
$Y = \langle B, D, C, A, B, A \rangle$

In the ***longest-common-subsequence problem***, we are given two sequences $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ and wish to find a maximum-length common subsequence of $X$ and $Y$. This section shows that the LCS problem can be solved efficiently using dynamic programming.

# ۳. Longest Common Subsequence(LCS)

we define the $i$th ***prefix*** of $X$, for $i = 0, 1, \ldots, m$, as $X_i = \langle x_1, x_2, \ldots, x_i \rangle$. For example, if $X = \langle A, B, C, B, D, A, B \rangle$, then $X_4 = \langle A, B, C, B \rangle$ and $X_0$ is the empty sequence.

***Theorem 15.1 (Optimal substructure of an LCS)***

Let $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ be sequences, and let $Z = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of $X$ and $Y$.

1. If $x_m = y_n$, then $z_k = x_m = y_n$ and $Z_{k-1}$ is an LCS of $X_{m-1}$ and $Y_{n-1}$.
2. If $x_m \neq y_n$, then $z_k \neq x_m$ implies that $Z$ is an LCS of $X_{m-1}$ and $Y$.
3. If $x_m \neq y_n$, then $z_k \neq y_n$ implies that $Z$ is an LCS of $X$ and $Y_{n-1}$.

# ۳. Longest Common Subsequence(LCS)

**Brute-Force ?**

Let us define $c[i, j]$ to be the length of an LCS of the sequences $X_i$ and $Y_j$. If either $i = 0$ or $j = 0$, one of the sequences has length 0, so the LCS has length 0. The optimal substructure of the LCS problem gives the recursive formula

$$c[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 , \\ c[i-1, j-1] + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j , \\ \max(c[i, j-1], c[i-1, j]) & \text{if } i, j > 0 \text{ and } x_i \neq y_j . \end{cases} \tag{15.14}$$

# ٣. Longest Common Subsequence(LCS)

$b[i, j] \leftarrow$ "$\nwarrow$" $\quad c[i, j] \leftarrow c[i-1, j-1] + 1$

$b[i, j] \leftarrow$ "$\uparrow$" $\quad c[i-1, j] \geq c[i, j-1]$

$b[i, j] \leftarrow$ "$\leftarrow$" $\quad c[i, j] \leftarrow c[i, j-1]$

# ۳. Longest Common Subsequence(LCS)

```
LCS-LENGTH(X, Y)
m ← length[X]
n ← length[Y]
for i ← 1 to m
        do c[i, 0] ← 0
for j ← 0 to n
        do c[0, j] ← 0
for i ← 1 to m
        do for j ← 1 to n
                do if x_i = y_j
                        then c[i, j] ← c[i − 1, j − 1] + 1
                             b[i, j] ← "↖"
                        else if c[i − 1, j] ≥ c[i, j − 1]
                                then c[i, j] ← c[i − 1, j]
                                     b[i, j] ← "↑"
                                else c[i, j] ← c[i, j − 1]
                                     b[i, j] ← "←"
return c and b
```

# ۳. Longest Common Subsequence(LCS)

| | $j$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| $i$ | | $y_j$ | $B$ | $D$ | $C$ | $A$ | $B$ | $A$ |
| 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 |

# ٣. Longest Common Subsequence(LCS)

```
PRINT-LCS(b, X, i, j)
if i = 0 or j = 0
    then return
if b[i, j] = "↖"
    then PRINT-LCS(b, X, i − 1, j − 1)
         print x_i
elseif b[i, j] = "↑"
    then PRINT-LCS(b, X, i − 1, j)
else PRINT-LCS(b, X, i, j − 1)
```

# ۳. Longest Common Subsequence(LCS)

***15.4-5***

Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of $n$ numbers.

***15.4-6*** ★

Give an $O(n \lg n)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of $n$ numbers. (*Hint:* Observe that the last element of a candidate subsequence of length $i$ is at least as large as the last element of a candidate subsequence of length $i - 1$. Maintain candidate subsequences by linking them through the input sequence.)