

Design and Analysis of Algorithms

Part II: Dynamic Programming

Lecture 7: Chain Matrix Multiplication, Longest Common Subsequence and Minimum Edit Distance



Yongxin Tong (童咏昕)

School of CSE, Beihang University

yxtong@buaa.edu.cn

Outline

- **Review to Part II**
- **Chain Matrix Multiplication Problem**
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- **Longest Common Subsequence Problem**
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- **Minimum Edit Distance Problem**
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Introduction to Part II

- In Part II, we will illustrate Dynamic Programming (DP) using several examples:
 - 0-1 Knapsack (0-1背包)
 - Rod-Cutting (钢条切割)
 - Chain Matrix Multiplication (矩阵链乘法)
 - Longest Common Subsequences (最长公共子序列)
 - Minimum Edit Distance (最小编辑距离)
 - All-Pairs Shortest Paths (所有结点对的最短路径)

Introduction to Part II

- In Part II, we will illustrate Dynamic Programming (DP) using several examples:
 - 0-1 Knapsack (0-1背包)
 - Rod-Cutting (钢条切割)
 - Chain Matrix Multiplication (矩阵链乘法)
 - Longest Common Subsequences (最长公共子序列)
 - Minimum Edit Distance (最小编辑距离)
 - All-Pairs Shortest Paths (所有结点对的最短路径)

Outline

- Review to Part II
- **Chain Matrix Multiplication Problem**
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Review of Matrix Multiplication

Matrix: An $n \times m$ matrix $A = [a[i, j]]$ is a two-dimensional array

$$A = \begin{bmatrix} a[1, 1] & a[1, 2] & \cdots & a[1, m - 1] & a[1, m] \\ a[2, 1] & a[2, 2] & \cdots & a[2, m - 1] & a[2, m] \\ \vdots & \vdots & & \vdots & \vdots \\ a[n, 1] & a[n, 2] & \cdots & a[n, m - 1] & a[n, m] \end{bmatrix},$$

which has n rows and m columns.

Review of Matrix Multiplication

Matrix: An $n \times m$ matrix $A = [a[i, j]]$ is a two-dimensional array

$$A = \begin{bmatrix} a[1, 1] & a[1, 2] & \cdots & a[1, m - 1] & a[1, m] \\ a[2, 1] & a[2, 2] & \cdots & a[2, m - 1] & a[2, m] \\ \vdots & \vdots & & \vdots & \vdots \\ a[n, 1] & a[n, 2] & \cdots & a[n, m - 1] & a[n, m] \end{bmatrix},$$

which has n rows and m columns.

Example

The following is a 4×5 matrix:

$$\begin{bmatrix} 12 & 8 & 9 & 7 & 6 \\ 7 & 6 & 89 & 56 & 2 \\ 5 & 5 & 6 & 9 & 10 \\ 8 & 6 & 0 & -8 & -1 \end{bmatrix}.$$

Review of Matrix Multiplication

The product $C = AB$ of a $p \times q$ matrix A and a $q \times r$ matrix B is a $p \times r$ matrix given by

$$c[i, j] = \sum_{k=1}^q a[i, k]b[k, j], \text{ for } 1 \leq i \leq p \text{ and } 1 \leq j \leq r$$

Review of Matrix Multiplication

The product $C = AB$ of a $p \times q$ matrix A and a $q \times r$ matrix B is a $p \times r$ matrix given by

$$c[i, j] = \sum_{k=1}^q a[i, k]b[k, j], \text{ for } 1 \leq i \leq p \text{ and } 1 \leq j \leq r$$

Complexity of Matrix multiplication: Note that C has pr entries and each entry takes $\Theta(q)$ time to compute so the total procedure takes $\Theta(pqr)$ time.

Review of Matrix Multiplication

The product $C = AB$ of a $p \times q$ matrix A and a $q \times r$ matrix B is a $p \times r$ matrix given by

$$c[i, j] = \sum_{k=1}^q a[i, k]b[k, j], \text{ for } 1 \leq i \leq p \text{ and } 1 \leq j \leq r$$

Complexity of Matrix multiplication: Note that C has pr entries and each entry takes $\Theta(q)$ time to compute so the total procedure takes $\Theta(pqr)$ time.

Example

$$A = \begin{bmatrix} 1 & 8 & 9 \\ 7 & 6 & -1 \\ 5 & 5 & 6 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 8 \\ 7 & 6 \\ 5 & 5 \end{bmatrix}, \quad C = AB = \begin{bmatrix} 102 & 101 \\ 44 & 87 \\ 70 & 100 \end{bmatrix}.$$

Remarks on Matrix Multiplication

- Matrix multiplication is **associative**, e.g.,

$$A_1 A_2 A_3 = (A_1 A_2) A_3 = A_1 (A_2 A_3)$$

so parenthesization does not change result.

Remarks on Matrix Multiplication

- Matrix multiplication is **associative**, e.g.,

$$A_1 A_2 A_3 = (A_1 A_2) A_3 = A_1 (A_2 A_3)$$

so parenthesization does not change result.

- Matrix multiplication is **NOT commutative**, e.g.,

$$A_1 A_2 \neq A_2 A_1$$

Matrix Multiplication of ABC

- Given a $p \times q$ matrix A, a $q \times r$ matrix B and a $r \times s$ matrix C, then ABC can be computed in two ways $(AB)C$ and $A(BC)$.

Matrix Multiplication of ABC

- Given a $p \times q$ matrix A, a $q \times r$ matrix B and a $r \times s$ matrix C, then ABC can be computed in two ways $(AB)C$ and $A(BC)$
- The number of multiplications needed are:

$$\text{mult}[(AB)C] = pqr + prs,$$

$$\text{mult}[A(BC)] = qrs + pqs.$$

Matrix Multiplication of ABC

- Given a $p \times q$ matrix A, a $q \times r$ matrix B and a $r \times s$ matrix C, then ABC can be computed in two ways $(AB)C$ and $A(BC)$
- The number of multiplications needed are:

$$\text{mult}[(AB)C] = pqr + prs,$$

$$\text{mult}[A(BC)] = qrs + pqs.$$

Example

When $p = 5$, $q = 4$, $r = 6$ and $s = 2$, then

$$\text{mult}[(AB)C] = 180,$$

$$\text{mult}[A(BC)] = 88.$$

A big difference!

important!!

Outline

- Review to Part II
- **Chain Matrix Multiplication Problem**
 - Review of Matrix Multiplication
 - **The Chain Matrix Multiplication Problem**
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

The Chain Matrix Multiplication Problem

Definition (Chain matrix multiplication problem)

Given dimensions p_0, p_1, \dots, p_n , corresponding to matrix sequence A_1, A_2, \dots, A_n where A_i has dimension $p_{i-1} \times p_i$, determine the “multiplication sequence” that minimizes the number of scalar multiplications in computing $A_1 A_2 \cdots A_n$.

The Chain Matrix Multiplication Problem

Definition (Chain matrix multiplication problem)

Given dimensions p_0, p_1, \dots, p_n , corresponding to matrix sequence A_1, A_2, \dots, A_n where A_i has dimension $p_{i-1} \times p_i$, determine the “multiplication sequence” that minimizes the number of scalar multiplications in computing $A_1 A_2 \cdots A_n$.

- i.e., determine how to parenthesize the multiplications.

Example

$$\begin{aligned}A_1 A_2 A_3 A_4 &= (A_1 A_2)(A_3 A_4) = A_1(A_2(A_3 A_4)) = A_1((A_2 A_3) A_4) \\&= ((A_1 A_2) A_3)(A_4) = (A_1(A_2 A_3))(A_4)\end{aligned}$$

The Chain Matrix Multiplication Problem

Definition (Chain matrix multiplication problem)

Given dimensions p_0, p_1, \dots, p_n , corresponding to matrix sequence A_1, A_2, \dots, A_n where A_i has dimension $p_{i-1} \times p_i$, determine the “multiplication sequence” that minimizes the number of scalar multiplications in computing $A_1 A_2 \cdots A_n$.

- i.e., determine how to parenthesize the multiplications.

Example

$$\begin{aligned}A_1 A_2 A_3 A_4 &= (A_1 A_2)(A_3 A_4) = A_1(A_2(A_3 A_4)) = A_1((A_2 A_3) A_4) \\&= ((A_1 A_2) A_3)(A_4) = (A_1(A_2 A_3))(A_4)\end{aligned}$$

Exhaustive search: $\Omega(4^n/n^{3/2})$.

Question

Any better approach?

Yes – DP

Outline

- Review to Part II
- **Chain Matrix Multiplication Problem**
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - **A Dynamic Programming Algorithm**
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

A Dynamic Programming Algorithm

Step 1: Space of subproblems

- For each pair $1 \leq i \leq j \leq n$, determine the multiplication sequence for $A_{i..j} = A_i A_{i+1} \cdots A_j$ that minimizes the number of multiplications.

A Dynamic Programming Algorithm

Step 1: Space of subproblems

- For each pair $1 \leq i \leq j \leq n$, determine the multiplication sequence for $A_{i..j} = A_i A_{i+1} \cdots A_j$ that minimizes the number of multiplications.
- Clearly, $A_{i..j}$ is a $p_{i-1} \times p_j$ matrix.

A Dynamic Programming Algorithm

Step 1: Space of subproblems

- For each pair $1 \leq i \leq j \leq n$, determine the multiplication sequence for $A_{i..j} = A_i A_{i+1} \cdots A_j$ that minimizes the number of multiplications.
- Clearly, $A_{i..j}$ is a $p_{i-1} \times p_j$ matrix.
- Original Problem: determine sequence of multiplication for $A_{1..n}$.

Relationships among Subproblems

For any optimal multiplication sequence, at the last step you are multiplying two matrices $A_{i..k}$ and $A_{k+1..j}$ for some k . That is,

$$A_{i..j} = (A_i \cdots A_k)(A_{k+1} \cdots A_j) = A_{i..k}A_{k+1..j}.$$

Relationships among Subproblems

For any optimal multiplication sequence, at the last step you are multiplying two matrices $A_{i..k}$ and $A_{k+1..j}$ for some k . That is,

$$A_{i..j} = (A_i \cdots A_k)(A_{k+1} \cdots A_j) = A_{i..k}A_{k+1..j}.$$

Question

How do we decide where to split the chain (what is k)?

ANS: Need to search all possible values of k .

Relationships among Subproblems

For any optimal multiplication sequence, at the last step you are multiplying two matrices $A_{i..k}$ and $A_{k+1..j}$ for some k . That is,

$$A_{i..j} = (A_i \cdots A_k)(A_{k+1} \cdots A_j) = A_{i..k}A_{k+1..j}.$$

Question

How do we decide where to split the chain (what is k)?

ANS: Need to search all possible values of k .

Question

How do we parenthesize the subchains $A_{i..k}$ and $A_{k+1..j}$?

ANS: $A_{i..k}$ and $A_{k+1..j}$ must be computed optimally, so we can apply the same procedure recursively.

Optimal Structure Property

If the final "optimal" solution of $A_{i..j}$ involves splitting into $A_{i..k}$ and $A_{k+1..j}$ at the final step, then parenthesization of $A_{i..k}$ and $A_{k+1..j}$ in the final optimal solution must also be **optimal** for the subproblems

Optimal Structure Property

If the final "optimal" solution of $A_{i..j}$ involves splitting into $A_{i..k}$ and $A_{k+1..j}$ at the final step, then parenthesization of $A_{i..k}$ and $A_{k+1..j}$ in the final optimal solution must also be **optimal** for the subproblems

- If parenthesization of $A_{i..k}$ was **not** optimal, we could replace it by a better parenthesization and get a cheaper final solution, leading to a contradiction.

Optimal Structure Property

If the final "optimal" solution of $A_{i..j}$ involves splitting into $A_{i..k}$ and $A_{k+1..j}$ at the final step, then parenthesization of $A_{i..k}$ and $A_{k+1..j}$ in the final optimal solution must also be **optimal** for the subproblems

- If parenthesization of $A_{i..k}$ was **not** optimal, we could replace it by a better parenthesization and get a cheaper final solution, leading to a contradiction.
- Similarly, if parenthesization of $A_{k+1..j}$ was **not** optimal, we could replace it by a better parenthesization and get a cheaper final solution, also leading to a contradiction.

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

- For $1 \leq i \leq j \leq n$, let $m[i, j]$ denote the minimum number of multiplications needed to compute $A_{i..j}$. The **optimum** cost can be described by the following recursive definition.

$$m[i, j] = \begin{cases} 0, & i = j \\ \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j), & i < j \end{cases}$$

Proof

Any optimal sequence of multiplication for $A_{i..j}$ is equivalent to some choice of splitting $A_{i..j} = A_{i..k}A_{k+1..j}$ for some k , where the sequences of multiplications for $A_{i..k}$ and $A_{k+1..j}$ are also optimal.

Proof

Any optimal sequence of multiplication for $A_{i..j}$ is equivalent to some choice of splitting $A_{i..j} = A_{i..k}A_{k+1..j}$ for some k , where the sequences of multiplications for $A_{i..k}$ and $A_{k+1..j}$ are also optimal.

Hence

$$m[i, j] = m[i, k] + m[k + 1, j] + p_{i-1}p_k p_j.$$

We don't know what k is, though.

Proof

Any optimal sequence of multiplication for $A_{i..j}$ is equivalent to some choice of splitting $A_{i..j} = A_{i..k}A_{k+1..j}$ for some k , where the sequences of multiplications for $A_{i..k}$ and $A_{k+1..j}$ are also optimal.

Hence

$$m[i, j] = m[i, k] + m[k + 1, j] + p_{i-1}p_kp_j.$$

We don't know what k is, though.

But, there are only $j - i$ possible values of k so we can check them all and find the one which returns a smallest cost.

Therefore

$$m[i, j] = \begin{cases} 0, & i = j \\ \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1}p_kp_j), & i < j \end{cases}$$

A Dynamic Programming Algorithm

Step 3: Bottom-up computation of $m[i, j]$

Recurrence:

$$m[i, j] = \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j)$$

A Dynamic Programming Algorithm

Step 3: Bottom-up computation of $m[i, j]$

Recurrence:

$$m[i, j] = \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1}p_kp_j)$$

Compute and save $m[i, j]$ in such an order that when it is time to calculate $m[i, j]$, the values of $m[i, k]$ and $m[k + 1, j]$ are already available.

A Dynamic Programming Algorithm

Step 3: Bottom-up computation of $m[i, j]$

Recurrence:

$$m[i, j] = \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j)$$

Compute and save $m[i, j]$ in such an order that when it is time to calculate $m[i, j]$, the values of $m[i, k]$ and $m[k + 1, j]$ are already available.

Compute them in increasing order of the length of the matrix-chain:

$m[1,2], m[2,3], m[3,4], \dots, m[n-3,n-2], m[n-2,n-1], m[n-1,n]$

A Dynamic Programming Algorithm

Step 3: Bottom-up computation of $m[i, j]$

Recurrence:

$$m[i, j] = \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j)$$

Compute and save $m[i, j]$ in such an order that when it is time to calculate $m[i, j]$, the values of $m[i, k]$ and $m[k + 1, j]$ are already available.

Compute them in increasing order of the length of the matrix-chain:

$m[1,2], m[2,3], m[3,4], \dots, m[n-3,n-2], m[n-2,n-1], m[n-1,n]$

$m[1,3], m[2,4], m[3,5], \dots, m[n-3,n-1], m[n-2,n]$

A Dynamic Programming Algorithm

Step 3: Bottom-up computation of $m[i, j]$

Recurrence:

$$m[i, j] = \min_{i \leq k < j} (m[i, k] + m[k + 1, j] + p_{i-1} p_k p_j)$$

Compute and save $m[i, j]$ in such an order that when it is time to calculate $m[i, j]$, the values of $m[i, k]$ and $m[k + 1, j]$ are already available.

Compute them in increasing order of the length of the matrix-chain:

$m[1,2], m[2,3], m[3,4], \dots, m[n-3,n-2], m[n-2,n-1], m[n-1,n]$

$m[1,3], m[2,4], m[3,5], \dots, m[n-3,n-1], m[n-2,n]$

$m[1,4], m[2,5], m[3,6], \dots, m[n-3,n] \dots$

$m[1,n-1], m[2,n]$

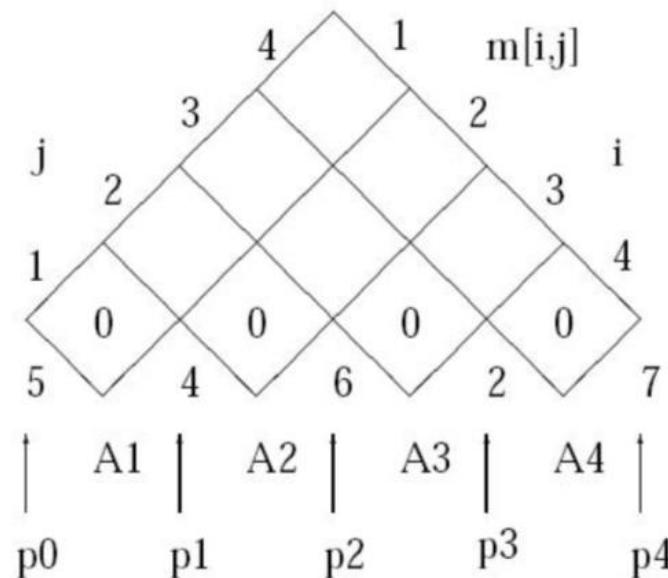
$m[1,n]$

Example for Bottom-Up Computation

Example

Given a chain of four matrices A_1, A_2, A_3 and A_4 , with $p_0 = 5$, $p_1 = 4$, $p_2 = 6$, $p_3 = 2$ and $p_4 = 7$. Find $m[1,4]$.

S0: Initialization



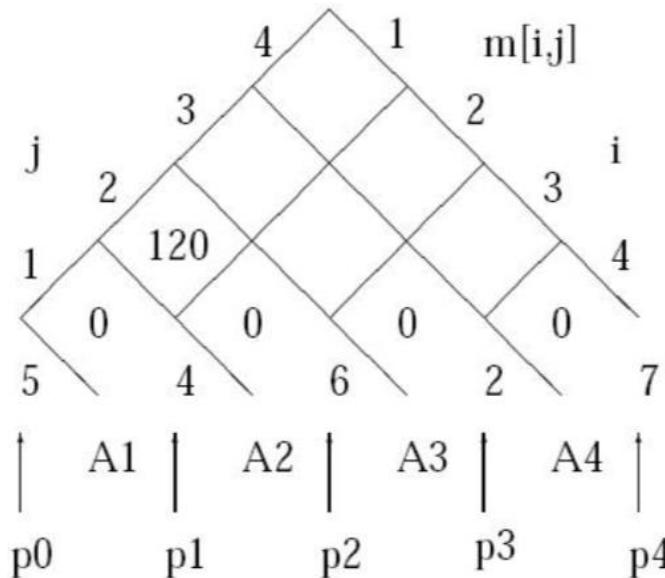
Example - Continued

Step 1: Computing $m[1,2]$

By definition

$$m[1,2] = \min_{1 \leq k < 2} (m[1,k] + m[k+1,2] + p_0 p_k p_2)$$

$$m[1,1] + m[2,2] + p_0 p_1 p_2 = 120$$

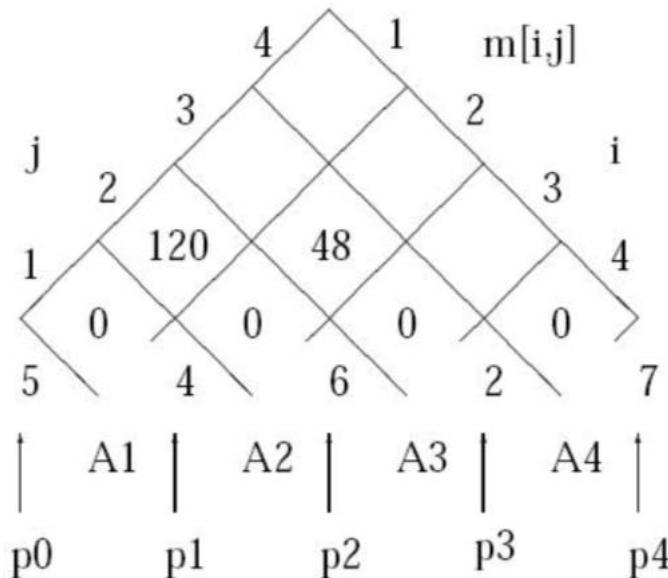


Example - Continued

Step 2: Computing $m[2,3]$

By definition

$$\begin{aligned} m[2,3] &= \min_{2 \leq k < 3} (m[2,k] + m[k+1,3] + p_1 p_k p_3) \\ &= m[2,2] + m[3,3] + p_1 p_2 p_3 = 48 \end{aligned}$$

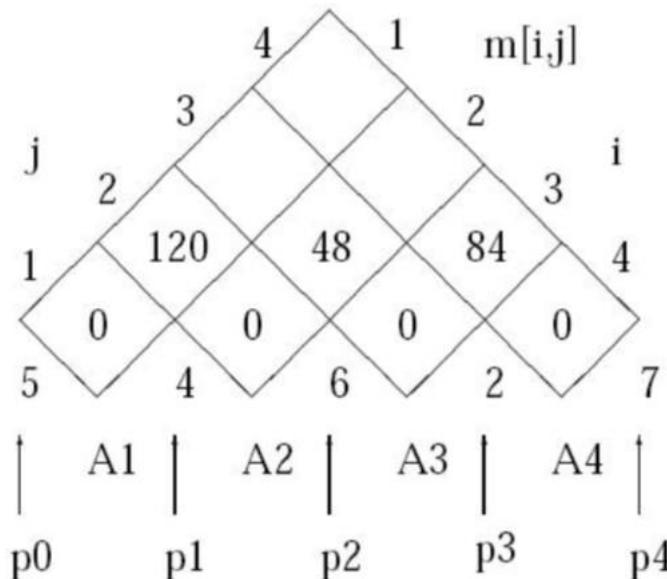


Example - Continued

Step 3: Computing $m[3,4]$

By definition

$$\begin{aligned} m[3,4] &= \min_{3 \leq k < 4} (m[3,k] + m[k+1,4] + p_2 p_k p_4) \\ &= m[3,3] + m[4,4] + p_2 p_3 p_4 = 84 \end{aligned}$$

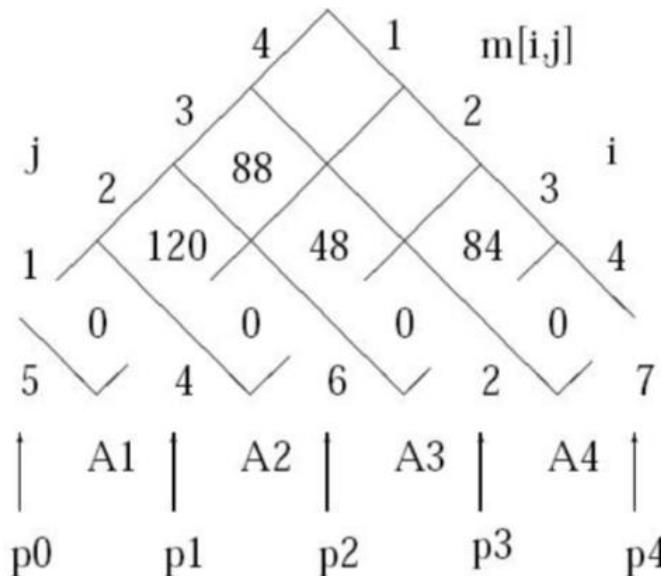


Example - Continued

Step 4: Computing $m[1,3]$

By definition

$$\begin{aligned}
 m[1,3] &= \min_{1 \leq k < 3} (m[1,k] + m[k+1,3] + p_0 p_k p_3) \\
 &= \min \left\{ \begin{array}{l} m[1,1] + m[2,3] + p_0 p_1 p_3 \\ m[1,2] + m[3,3] + p_0 p_2 p_3 \end{array} \right. \\
 &= 88
 \end{aligned}$$

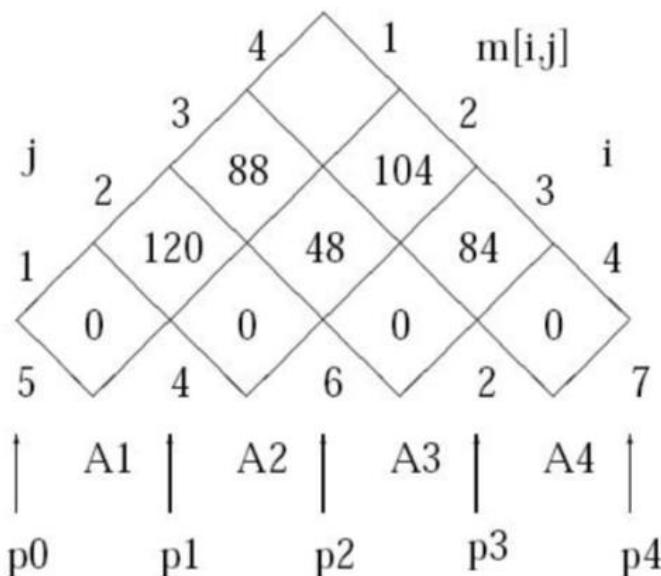


Example - Continued

Step 5: Computing $m[2,4]$

By definition

$$\begin{aligned} m[2,4] &= \min_{2 \leq k < 4} (m[2,k] + m[k+1,4] + p_1 p_k p_4) \\ &= \min \left\{ \begin{array}{l} m[2,2] + m[3,4] + p_1 p_2 p_4 \\ m[2,3] + m[4,4] + p_1 p_3 p_4 \end{array} \right. \\ &= 104 \end{aligned}$$

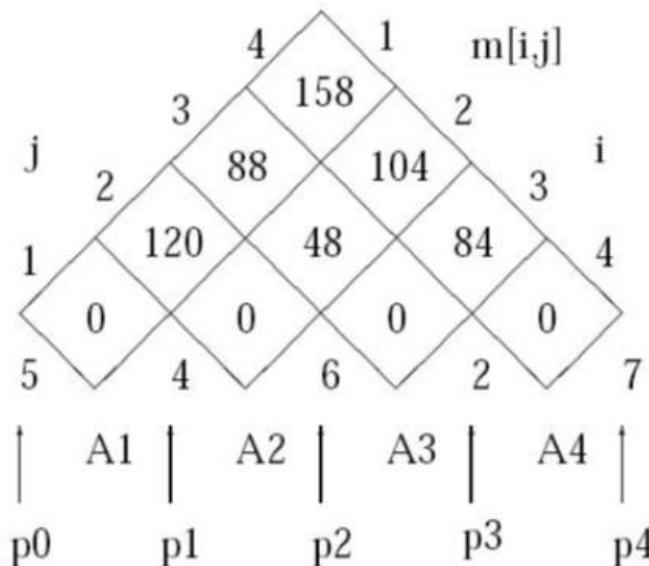


Example - Continued

Step 6: Computing $m[1,4]$

By definition

$$\begin{aligned} m[1,4] &= \min_{1 \leq k < 4} (m[1,k] + m[k+1,4] + p_0 p_k p_4) \\ &= \min \begin{cases} m[1,1] + m[2,4] + p_0 p_1 p_4 \\ m[1,2] + m[3,4] + p_0 p_2 p_4 \\ m[1,3] + m[4,4] + p_0 p_3 p_4 \end{cases} \\ &= 158 \end{aligned}$$



A Dynamic Programming Algorithm

Step 4: Constructing optimal solution

Idea: Maintain an array $s[1..n, 1..n]$, where $s[i,j]$ denotes k for the optimal splitting in computing $A_{i..j} = A_{i..k}A_{k+1..j}$

Question

How to Recover the Multiplication Sequence using $s[1..n, 1..n]$?

A Dynamic Programming Algorithm

Step 4: Constructing optimal solution

Idea: Maintain an array $s[1..n, 1..n]$, where $s[i,j]$ denotes k for the optimal splitting in computing $A_{i..j} = A_{i..k}A_{k+1..j}$

Question

How to Recover the Multiplication Sequence using $s[1..n, 1..n]$?

$$\begin{array}{ll}
 s[1, n] & (A_1 \cdots A_{s[1,n]})(A_{s[1,n]+1} \cdots A_n) \\
 s[1, s[1, n]] & (A_1 \cdots A_{s[1,s[1,n]]})(A_{s[1,s[1,n]]+1} \cdots A_{s[1,n]}) \\
 s[s[1, n] + 1, n] & (A_{s[1,n]+1} \cdots A_{s[s[1,n]+1,n]})(A_{s[s[1,n]+1,n]+1} \cdots A_n) \\
 \vdots & \vdots
 \end{array}$$

Do this **recursively** until the multiplication sequence is determined.

An Example of Step 4

Example (Finding the Multiplication Sequence)

Consider $n = 6$. Assume that the array $s[1..6, 1..6]$ has been computed. The multiplication sequence is recovered as follows.

An Example of Step 4

Example (Finding the Multiplication Sequence)

Consider $n = 6$. Assume that the array $s[1..6, 1..6]$ has been computed. The multiplication sequence is recovered as follows.

$$s[1, 6] = 3 \quad (A_1 A_2 A_3)(A_4 A_5 A_6)$$

$$s[1, 3] = 1 \quad (A_1 (A_2 A_3))$$

$$s[4, 6] = 5 \quad ((A_4 A_5) A_6)$$

An Example of Step 4

Example (Finding the Multiplication Sequence)

Consider $n = 6$. Assume that the array $s[1..6, 1..6]$ has been computed. The multiplication sequence is recovered as follows.

$$s[1, 6] = 3 \quad (A_1 A_2 A_3)(A_4 A_5 A_6)$$

$$s[1, 3] = 1 \quad (A_1(A_2 A_3))$$

$$s[4, 6] = 5 \quad ((A_4 A_5) A_6)$$

Hence the final multiplication sequence is

$$(A_1(A_2 A_3))((A_4 A_5) A_6).$$

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

```
for  $i \leftarrow 1$  to  $n$  do  
|  $m[i, i] \leftarrow 0$ ;  
end
```

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

for $i \leftarrow 1$ to n **do**

 | $m[i, i] \leftarrow 0;$

end

for $l \leftarrow 2$ to n **do**

 | **for** $i \leftarrow 1$ to $n - l + 1$ **do**

 | $j \leftarrow i + l - 1;$

 | :

 | :

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

for $i \leftarrow 1$ to n **do**

 | $m[i, i] \leftarrow 0$;

end

for $l \leftarrow 2$ to n **do**

 | **for** $i \leftarrow 1$ to $n - l + 1$ **do**

 | $j \leftarrow i + l - 1$;

 | $m[i, j] \leftarrow \infty$;

 |

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

for $i \leftarrow 1$ to n **do**

 | $m[i, i] \leftarrow 0$;

end

for $l \leftarrow 2$ to n **do**

 | **for** $i \leftarrow 1$ to $n - l + 1$ **do**

 | $j \leftarrow i + l - 1$;

 | $m[i, j] \leftarrow \infty$;

 | **for** $k \leftarrow i$ to $j - 1$ **do**

 | $q \leftarrow m[i, k] + m[k + 1, j] + p[i - 1] * p[k] * p[j]$;

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

```
for  $i \leftarrow 1$  to  $n$  do
    |  $m[i, i] \leftarrow 0$ ;
end
for  $l \leftarrow 2$  to  $n$  do
    for  $i \leftarrow 1$  to  $n - l + 1$  do
        |  $j \leftarrow i + l - 1$ ;
        |  $m[i, j] \leftarrow \infty$ ;
        for  $k \leftarrow i$  to  $j - 1$  do
            |  $q \leftarrow m[i, k] + m[k + 1, j] + p[i - 1] * p[k] * p[j]$ ;
            | if  $q < m[i, j]$  then
                |     |  $m[i, j] \leftarrow q$ ;
                |     |  $s[i, j] \leftarrow k$ ;
                | end
            end
        end
    end
end
return
```

The Dynamic Programming Algorithm

MatrixChain(p, n)

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

```
for  $i \leftarrow 1$  to  $n$  do
    |  $m[i, i] \leftarrow 0$ ;
end
for  $l \leftarrow 2$  to  $n$  do
    for  $i \leftarrow 1$  to  $n - l + 1$  do
        |  $j \leftarrow i + l - 1$ ;
        |  $m[i, j] \leftarrow \infty$ ;
        for  $k \leftarrow i$  to  $j - 1$  do
            |  $q \leftarrow m[i, k] + m[k + 1, j] + p[i - 1] * p[k] * p[j]$ ;
            | if  $q < m[i, j]$  then
                |     |  $m[i, j] \leftarrow q$ ;
                |     |  $s[i, j] \leftarrow k$ ;
                | end
            end
        end
    end
return  $m[1, n]$  and  $s$ ;
```

The Dynamic Programming Algorithm

`MatrixChain(p, n)`

Let $m[1..n, 1..n]$ and $s[1..n, 1..n]$ be two 2-dimension arrays;

```

for  $i \leftarrow 1$  to  $n$  do
|    $m[i, i] \leftarrow 0$ ;
end
for  $l \leftarrow 2$  to  $n$  do
    for  $i \leftarrow 1$  to  $n - l + 1$  do
         $j \leftarrow i + l - 1$ ;
         $m[i, j] \leftarrow \infty$ ;
        for  $k \leftarrow i$  to  $j - 1$  do
             $q \leftarrow m[i, k] + m[k + 1, j] + p[i - 1] * p[k] * p[j]$ ;
            if  $q < m[i, j]$  then
                 $m[i, j] \leftarrow q$ ;
                 $s[i, j] \leftarrow k$ ;
            end
        end
    end
end
return  $m[1, n]$  and  $s$ ;
```

Complexity: The loops are nested three levels deep. Each loop index takes on $\leq n$ values. Hence the **time complexity** is $O(n^3)$. **Space complexity** is $\Theta(n^2)$.

Outline

- Review to Part II
- Chain Matrix Multiplication Problem
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Longest Common Subsequence

Given two sequences $X = (x_1, x_2, \dots, x_m)$ and $Y = (y_1, y_2, \dots, y_n)$, we say that Z is a *common subsequence* of X and Y if Z has a strictly increasing sequence of indices i and j of both X and Y such that we have $x_{i_p} = y_{j_p} = z_p$ for all $p = 1, 2, \dots, k$.

For example:

X : A B C B D A B

Y : B D C A B A

Longest Common Subsequence

Given two sequences $X = (x_1, x_2, \dots, x_m)$ and $Y = (y_1, y_2, \dots, y_n)$, we say that Z is a *common subsequence* of X and Y if Z has a strictly increasing sequence of indices i and j of both X and Y such that we have $x_{i_p} = y_{j_p} = z_p$ for all $p = 1, 2, \dots, k$.

For example:

X : A B C B D A B

Y : B D C A B A

We want to find a *longest common subsequence* of both X and Y .

Longest Common Subsequence

Given two sequences $X = (x_1, x_2, \dots, x_m)$ and $Y = (y_1, y_2, \dots, y_n)$, we say that Z is a *common subsequence* of X and Y if Z has a strictly increasing sequence of indices i and j of both X and Y such that we have $x_{i_p} = y_{j_p} = z_p$ for all $p = 1, 2, \dots, k$.

For example:

$X :$	A	B	C	B	D	A	B
$Y :$		B	D	C	A	B	A
$Z :$		B	C	B		A	

We want to find a *longest common subsequence* of both X and Y .

The longest common subsequence of X and Y is Z .

A Dynamic Programming Algorithm

Step 1: Space of subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d_{i,j}$ to be the length of the longest common subsequence of $X[1..i]$ and $Y [1..j]$.

A Dynamic Programming Algorithm

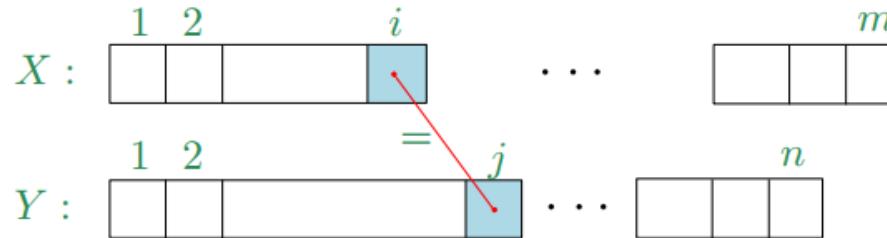
Step 1: Space of subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d_{i,j}$ to be the length of the longest common subsequence of $X[1..i]$ and $Y [1..j]$.
- Let D be the $m \times n$ matrix $[d_{i,j}]$.

Relationships among Subproblems

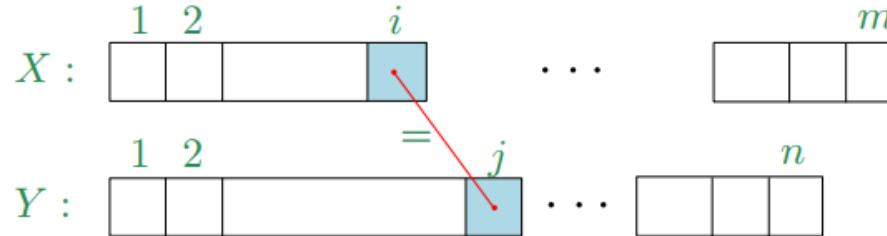
Step 2: Relating the value of a problem and those of its subproblems



Let $Z_k = (z_1, \dots, z_k)$ be a LCS of $X[1..i]$ and $Y [1..j]$.

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

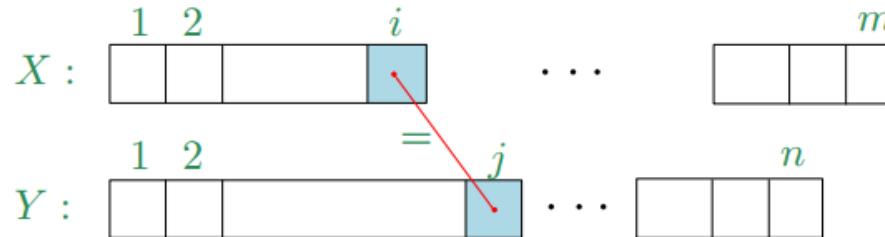


Let $Z_k = (z_1, \dots, z_k)$ be a LCS of $X[1..i]$ and $Y[1..j]$.

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of $X[1..i-1]$ and $Y[1..j-1]$.

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems



Let $Z_k = (z_1, \dots, z_k)$ be a LCS of $X[1..i]$ and $Y[1..j]$.

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of $X[1..i-1]$ and $Y[1..j-1]$.

Case 2: If $x_i \neq y_j$, it means that either the LCS does not end with x_i or does not end with y_j . Then Z_k is either a LCS of $X[1..i-1]$ and $Y[1..j]$, or a LCS of $X[1..i]$ and $Y[1..j-1]$. We carry on with the larger LCS count from either case.

$$d_{i,j} = \begin{cases} d_{i-1,j-1} + 1, & \text{if } x_i = y_j \\ \max\{d_{i-1,j}, d_{i,j-1}\}, & \text{if } x_i \neq y_j \end{cases}$$

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

Initially, we setup the first row and column of the matrix $d[0,j]$ and $d[i, 0]$ to 0

$D[i, j]$	$j = 0$	1	2	3	\dots	\dots	n
$i = 0$	0	0	0	0	\dots	\dots	0
1	0						
2	0						
\vdots	0						
m	0						

bottom
↓
up

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

Initially, we setup the first row and column of the matrix $d[0,j]$ and $d[i, 0]$ to 0

Calculate $d[i, j]$ for $j = 1, 2, \dots, n$

Then, the $d[i, j]$ for $i = 1, 2, \dots, m$

$D[i, j]$	$j = 0$	1	2	3	\dots	\dots	n
$i = 0$	0	0	0	0	\dots	\dots	0
1	0						
2	0						
\vdots	0						
m	0						

bottom
↓
up

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

Initially, we setup the first row and column of the matrix $d[0,j]$ and $d[i, 0]$ to 0

Calculate $d[i, j]$ for $j = 1, 2, \dots, n$

Then, the $d[i, j]$ for $i = 1, 2, \dots, m$

So, we fill the following table row by row and left to right.

$D[i, j]$	$j = 0$	1	2	3	\dots	\dots	n
$i = 0$	0	0	0	0	\dots	\dots	0
1	0						
2	0						
\vdots	0						
m	0						

bottom
↓
up

A Dynamic Programming Algorithm

Step 4: Construction of Optimal Solution

Also, we create another $m \times n$ matrix $p[i, j]$ for $1 \leq i \leq m$, and $1 \leq j \leq n$, to store arrows pointing to the element that was used in the computation. Thus, we can reconstruct the elements of an LCS later on.

	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	$\leftarrow 1$	1
2	B	0	1	$\leftarrow 1$	$\leftarrow 1$	1	2	$\leftarrow 2$
3	C	0	1	1	2	$\leftarrow 2$	2	2
4	B	0	1	1	2	2	3	$\leftarrow 3$
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

The Dynamic Programming Algorithm

Longest-Common-Subsequence(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Longest common subsequence of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ and $p[0..m, 0..n]$ be two new 2-dimension arrays;

The Dynamic Programming Algorithm

Longest-Common-Subsequence(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Longest common subsequence of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ and $p[0..m, 0..n]$ be two new 2-dimension arrays;

//Initialization

for $i \leftarrow 0$ to m **do**

 | $d[i, 0] \leftarrow 0;$

end

for $j \leftarrow 0$ to n **do**

 | $d[0, j] \leftarrow 0;$

end

The Dynamic Programming Algorithm

```
//Dynamic Programming
for i ← 1 to m do
    for j ← 1 to n do
        if  $x_i$  is equal to  $y_j$  then
             $d[i, j] \leftarrow d[i - 1, j - 1] + 1;$ 
             $p[i, j] \leftarrow "LU";$  // "LU" indicates left up arrow.
        end
```

The Dynamic Programming Algorithm

```
//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i$  is equal to  $y_j$  then
             $d[i, j] \leftarrow d[i - 1, j - 1] + 1;$ 
             $p[i, j] \leftarrow "LU";$  // "LU" indicates left up arrow.
        end
        else if  $d[i - 1, j] \geq d[i, j - 1]$  then
             $d[i, j] \leftarrow d[i - 1, j];$ 
             $p[i, j] \leftarrow "U";$  // "U" indicates up arrow.
        end
```

The Dynamic Programming Algorithm

```
//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i$  is equal to  $y_j$  then
             $d[i, j] \leftarrow d[i - 1, j - 1] + 1;$ 
             $p[i, j] \leftarrow "LU";$  // "LU" indicates left up arrow.
        end
        else if  $d[i - 1, j] \geq d[i, j - 1]$  then
             $d[i, j] \leftarrow d[i - 1, j];$ 
             $p[i, j] \leftarrow "U";$  // "U" indicates up arrow.
        end
        else
             $d[i, j] \leftarrow d[i, j - 1];$ 
             $p[i, j] \leftarrow "L";$  // "L" indicates left arrow.
        end
    end
end
return  $d, p;$ 
```

The Dynamic Programming Algorithm

```

//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i$  is equal to  $y_j$  then
             $d[i, j] \leftarrow d[i - 1, j - 1] + 1;$ 
             $p[i, j] \leftarrow "LU";$  // "LU" indicates left up arrow.
        end
        else if  $d[i - 1, j] \geq d[i, j - 1]$  then
             $d[i, j] \leftarrow d[i - 1, j];$ 
             $p[i, j] \leftarrow "U";$  // "U" indicates up arrow.
        end
        else
             $d[i, j] \leftarrow d[i, j - 1];$ 
             $p[i, j] \leftarrow "L";$  // "L" indicates left arrow.
        end
    end
end
return  $d, p;$ 

```

Obviously, the dynamic programming algorithm runs in $O(mn)$ time.

An Algorithm for Constructing Optimal Solutions

Print-LCS(p, X, i, j)

Input: Array p generated from Longest-Common-Subsequence, string X , index i and j .

Output: Output the longest common subsequence of $X[1..i]$ and $Y[1..j]$.

if i is equal to 0 or j is equal to 0 **then**

| **return** NULL;

end

An Algorithm for Constructing Optimal Solutions

Print-LCS(p, X, i, j)

Input: Array p generated from Longest-Common-Subsequence, string X , index i and j .

Output: Output the longest common subsequence of $X[1..i]$ and $Y[1..j]$.

if i is equal to 0 or j is equal to 0 **then**

 | **return** NULL;

end

if $p[i, j]$ is equal to "LU" **then**

 | Print-LCS($p, X, i - 1, j - 1$);

 | print x_i ;

end

An Algorithm for Constructing Optimal Solutions

Print-LCS(p, X, i, j)

Input: Array p generated from Longest-Common-Subsequence, string X , index i and j .

Output: Output the longest common subsequence of $X[1..i]$ and $Y[1..j]$.

```
if  $i$  is equal to 0 or  $j$  is equal to 0 then
    | return NULL;
end
if  $p[i, j]$  is equal to "LU" then
    | Print-LCS( $p, X, i - 1, j - 1$ );
    | print  $x_i$ ;
end
else if  $p[i, j]$  is equal to "U" then
    | Print-LCS( $p, X, i - 1, j$ );
end
else
    | Print-LCS( $p, X, i, j - 1$ );
end
```

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d



0

1

2

3

4

5

6

7

p



0

1

2

3

4

5

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1 0

2 0

3 0

4 0

5 0

6 0

7 0

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1

2

3

4

5

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						

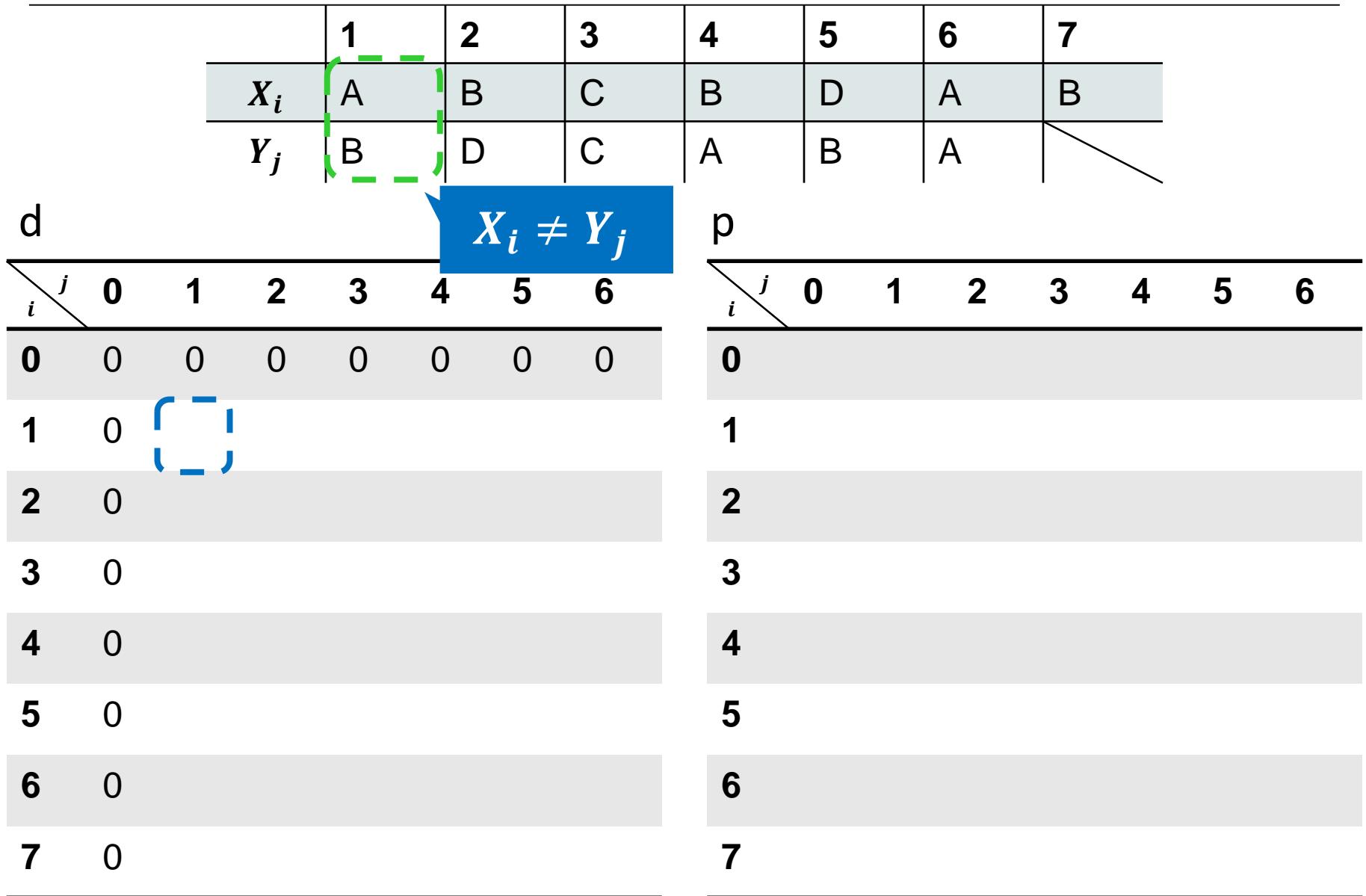
Initialization



1	0						
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

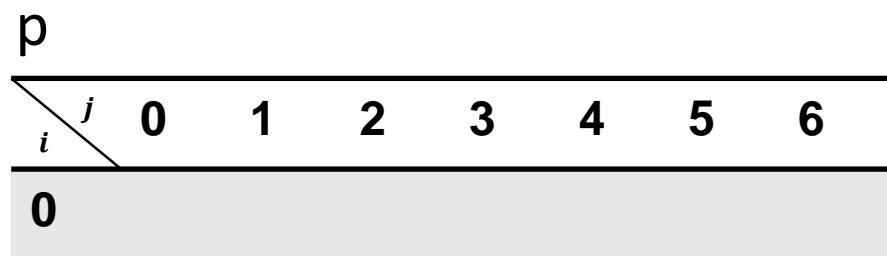
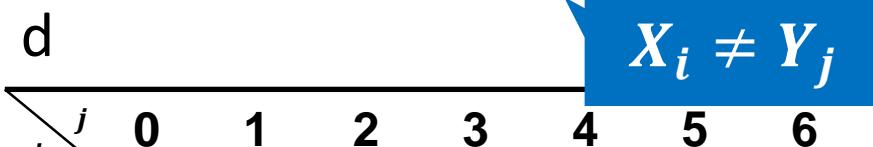
2							
3							
4							
5							
6							
7							

Example of Optimal Solution Construction



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



$$\max\{d[0, 1], d[1, 0]\}$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	U	U
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						
1	U	U					
2							
3							
4							
5							
6							
7							

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

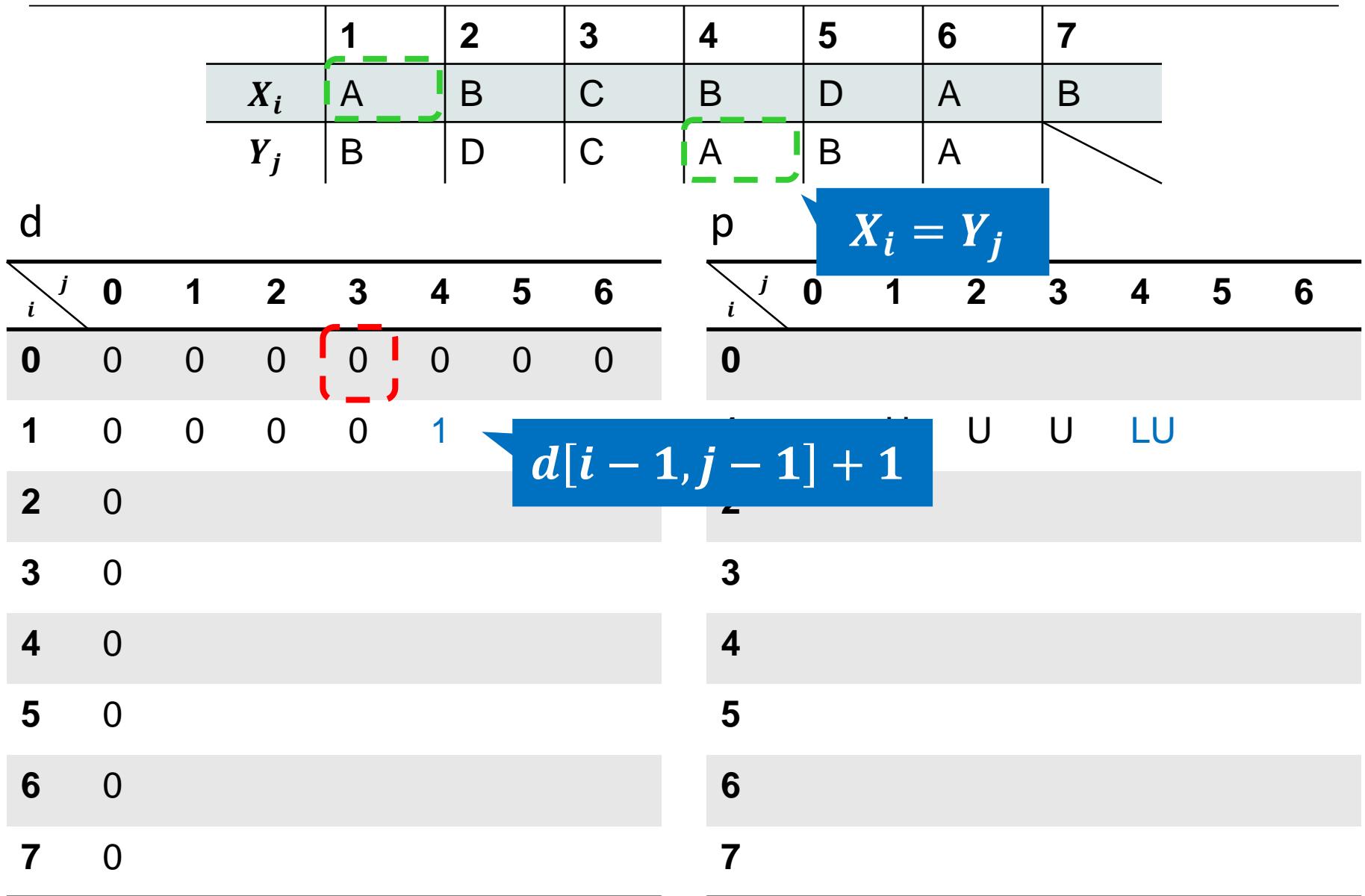
d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	U	U
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						
1	U	U	U	U			
2							
3							
4							
5							
6							
7							

Example of Optimal Solution Construction



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						
1	U	U	U	LU	L		
2							
3							
4							
5							
6							
7							

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1
2	0						
3	0						
4	0						
5	0						
6	0						

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						
1	1	U	U	U	LU	L	LU
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1					
---	---	---	--	--	--	--	--

3	0						
---	---	--	--	--	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	
---	---	---	---	----	---	----	--

2	LU						
---	----	--	--	--	--	--	--

3							
---	--	--	--	--	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1				
---	---	---	---	--	--	--	--

3	0						
---	---	--	--	--	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	
---	---	---	---	----	---	----	--

2	LU	L					
---	----	---	--	--	--	--	--

3							
---	--	--	--	--	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1			
---	---	---	---	---	--	--	--

3	0						
---	---	--	--	--	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L			
---	----	---	---	--	--	--

3						
---	--	--	--	--	--	--

4						
---	--	--	--	--	--	--

5						
---	--	--	--	--	--	--

6						
---	--	--	--	--	--	--

7						
---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	1	
---	---	---	---	---	---	---	--

3	0
---	---

4	0
---	---

5	0
---	---

6	0
---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U		
---	----	---	---	---	--	--

3	
---	--

4	
---	--

5	
---	--

6	
---	--

7	
---	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	
---	---	---	---	---	---	---	--

3	0
---	---

4	0
---	---

5	0
---	---

6	0
---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	
---	----	---	---	---	----	--

3

4						
---	--	--	--	--	--	--

5

6						
---	--	--	--	--	--	--

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
2	0	1	1	1	1	2	2

3	0
4	0
5	0
6	0
7	0

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	LU
2	LU	L	L	U	LU	L	

3	
4	
5	
6	
7	

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1					
---	---	---	--	--	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	
---	---	---	---	----	---	----	--

2	LU	L	L	U	LU	L	
---	----	---	---	---	----	---	--

3	U						
---	---	--	--	--	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	1			
---	---	---	---	---	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	LU
---	---	---	---	----	---	----	----

2	LU	L	L	U	LU	L	
---	----	---	---	---	----	---	--

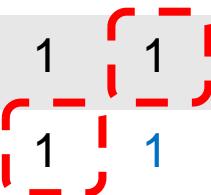
3	U	U					
---	---	---	--	--	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2			
---	---	---	---	---	--	--	--

4	0						
---	---	--	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU			
---	--	---	---	----	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	1	2	2
---	---	---	---	---	---	---	---	---

3	0	1	1	2	2	2		
---	---	---	---	---	---	---	--	--

4	0							
---	---	--	--	--	--	--	--	--

5	0							
---	---	--	--	--	--	--	--	--

6	0							
---	---	--	--	--	--	--	--	--

7	0							
---	---	--	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	
---	---	---	---	----	---	----	--

2	LU	L	L	U	LU	L	
---	----	---	---	---	----	---	--

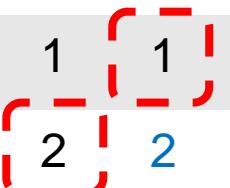
3	U	U	LU	L			
---	---	---	----	---	--	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2	2
---	---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2	2
---	---	---	---	---	---	---	---	---

4	0							
---	---	--	--	--	--	--	--	--

5	0							
---	---	--	--	--	--	--	--	--

6	0							
---	---	--	--	--	--	--	--	--

7	0							
---	---	--	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	LU
---	---	---	---	----	---	----	----

2	LU	L	L	U	LU	L	LU
---	----	---	---	---	----	---	----

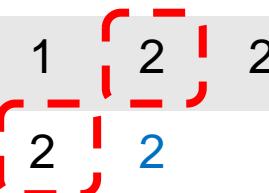
3	U	U	LU	L	U		
---	---	---	----	---	---	--	--

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--



2	2
---	---

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2	2
---	---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2	2
---	---	---	---	---	---	---	---	---

4	0							
---	---	--	--	--	--	--	--	--

5	0							
---	---	--	--	--	--	--	--	--

6	0							
---	---	--	--	--	--	--	--	--

7	0							
---	---	--	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

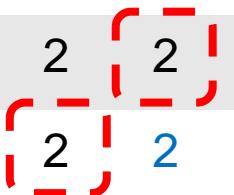
3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4							
---	--	--	--	--	--	--	--

5							
---	--	--	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1					
---	---	---	--	--	--	--	--

5	0						
---	---	--	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU					
---	----	--	--	--	--	--

5						
---	--	--	--	--	--	--

6						
---	--	--	--	--	--	--

7						
---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1 0 0 0 0 1 1 1

2 0 1 1 1 1 2 2

3 0 1 1 1 2 2 2

4 0 1 1 1

5 0

6 0

7 0

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						

1 U U U LU L LU

2 LU L L U LU L

3 U U LU L U U

4 LU U

5

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1 0 0 0 0 1 1 1

2 0 1 1 1 1 2 2

3 0 1 1 2 2 2 2

4 0 1 1 2

5 0

6 0

7 0

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1 U U U LU L LU

2 LU L L U LU L

3 U U LU L U U

4 LU U U

5

6

7



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

5	0
---	---

6	0
---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU	LU
---	---	---	---	----	---	----	----

2	LU	L	L	U	LU	L	LU
---	----	---	---	---	----	---	----

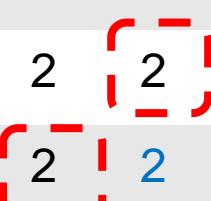
3	U	U	LU	L	U	U	U
---	---	---	----	---	---	---	---

4	LU	U	U	U	U	U	U
---	----	---	---	---	---	---	---

5

6

7



2

2

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3
---	---	---	---	---	---	---

5	0
---	---

6	0
---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU	U	U	U	U	LU
---	----	---	---	---	---	----

5

6

7



3

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0
---	---

6	0
---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

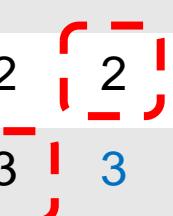
3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU	U	U	U	LU	L
---	----	---	---	---	----	---

5

6

7



3	2
---	---

4

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1					
---	---	---	--	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U					
---	--	---	--	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2				
---	---	---	---	--	--	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU				
---	--	---	----	--	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2		
---	---	---	---	---	---	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U			
---	--	---	----	---	--	--	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2		
---	---	---	---	---	---	--	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U		U	
---	--	---	----	---	--	---	--

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	
---	---	---	---	---	---	---	--

6	0						
---	---	--	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	1
2	0	1	1	1	1	2	2
3	0	1	1	2	2	2	2
4	0	1	1	2	2	3	3
5	0	1	2	2	2	3	3
6	0						
7	0						

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						
1	1	U	U	U	LU	L	LU
2	2	LU	L	L	U	LU	L
3	3	U	U	LU	L	U	U
4	4	LU	U	U	U	LU	L
5	5	U	LU	U	U	U	U
6	6						
7	7						

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1					
---	---	---	--	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U					
---	--	---	--	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2				
---	---	---	---	--	--	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U				
---	--	---	---	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	2		
---	---	---	---	---	---	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U				
---	--	---	---	--	--	--	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3		
---	---	---	---	---	---	--	--

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U		LU	
---	--	---	---	---	--	----	--

7							
---	--	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	3
---	---	---	---	---	---	---	---

7	0						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU	U	U	U	LU	L
---	----	---	---	---	----	---

5	U	LU	U	U	U	U
---	---	----	---	---	---	---

6	U	U	U	LU	U	U
---	---	---	---	----	---	---

7						
---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

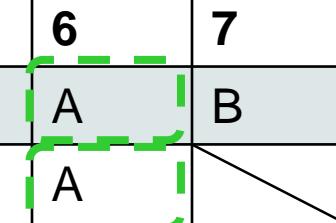
3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1					
---	---	---	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU	U	U	U	LU	L
---	----	---	---	---	----	---

5	U	LU	U	U	U	U
---	---	----	---	---	---	---

6	U	U	U	LU	U	LU
---	---	---	---	----	---	----

7	LU					
---	----	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2				
---	---	---	---	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7		LU	U				
---	--	----	---	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2	2	2		
---	---	---	---	---	---	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7		LU	U	U			
---	--	----	---	---	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2	2	3		
---	---	---	---	---	---	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7		LU	U	U	U		U
---	--	----	---	---	---	--	---

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2	2	3	4	
---	---	---	---	---	---	---	--

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7		LU	U	U	U	LU	
---	--	----	---	---	---	----	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2	2	3	4	4
---	---	---	---	---	---	---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0							

1		U	U	U	LU	L	LU
---	--	---	---	---	----	---	----

2		LU	L	L	U	LU	L
---	--	----	---	---	---	----	---

3		U	U	LU	L	U	U
---	--	---	---	----	---	---	---

4		LU	U	U	U	LU	L
---	--	----	---	---	---	----	---

5		U	LU	U	U	U	U
---	--	---	----	---	---	---	---

6		U	U	U	LU	U	LU
---	--	---	---	---	----	---	----

7		LU	U	U	U	LU	U
---	--	----	---	---	---	----	---

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0

1	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

2	0	1	1	1	1	2	2
---	---	---	---	---	---	---	---

3	0	1	1	2	2	2	2
---	---	---	---	---	---	---	---

4	0	1	1	2	2	3	3
---	---	---	---	---	---	---	---

5	0	1	2	2	2	3	3
---	---	---	---	---	---	---	---

6	0	1	2	2	3	3	4
---	---	---	---	---	---	---	---

7	0	1	2	2	3	4	4
---	---	---	---	---	---	---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0	0						

1	U	U	U	LU	L	LU
---	---	---	---	----	---	----

2	LU	L	L	U	LU	L
---	----	---	---	---	----	---

3	U	U	LU	L	U	U
---	---	---	----	---	---	---

4	LU	U	U	U	LU	L
---	----	---	---	---	----	---

5	U	LU	U	U	U	U
---	---	----	---	---	---	---

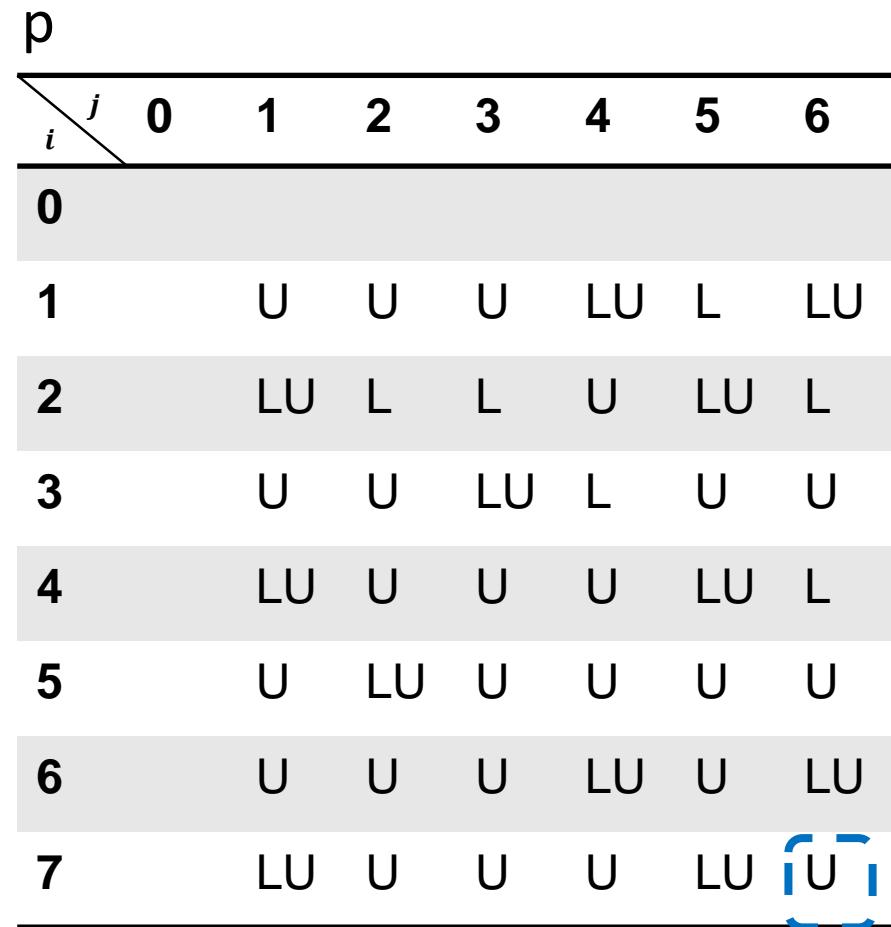
6	LU	U	LU			
---	----	---	----	--	--	--

7	U	LU	U			
---	---	----	---	--	--	--

Length of Longest
Common
Subsequence

Example of Optimal Solution Construction

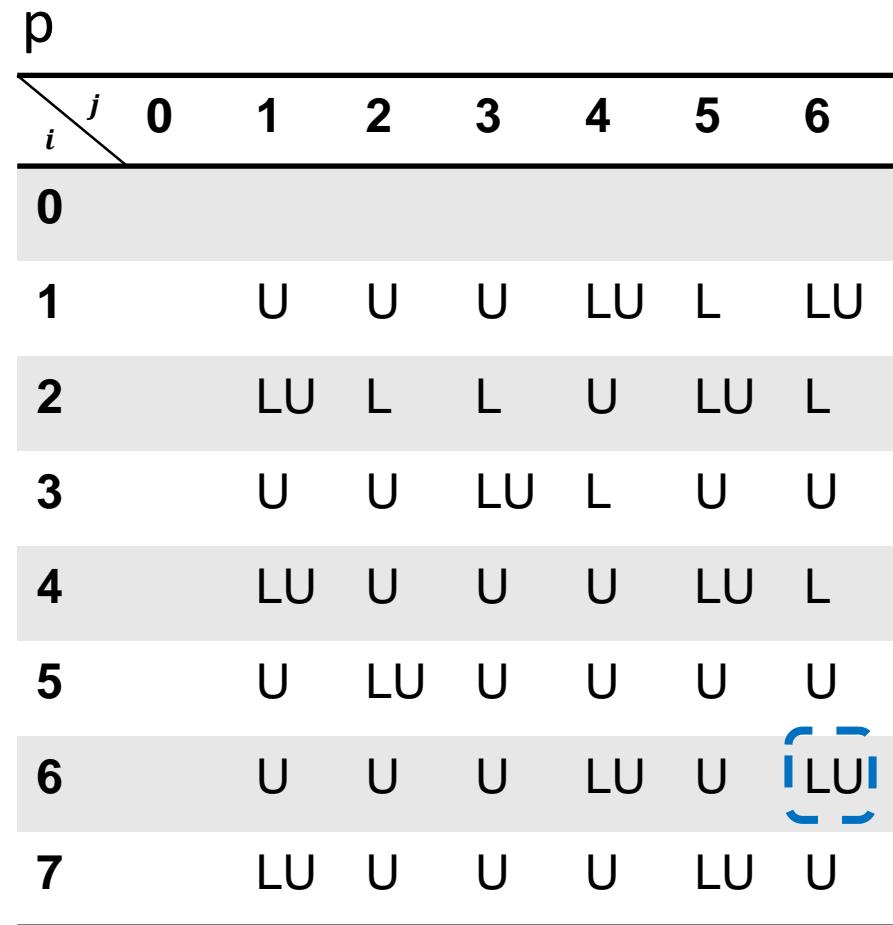
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = <>

Example of Optimal Solution Construction

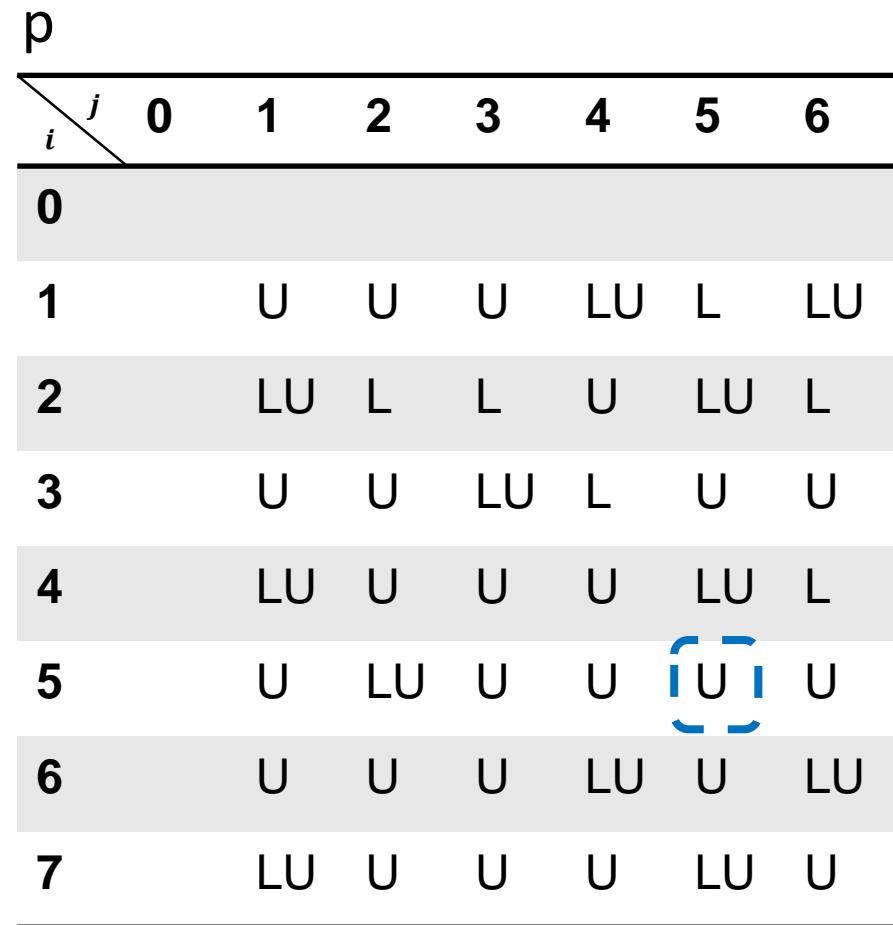
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = < A >

Example of Optimal Solution Construction

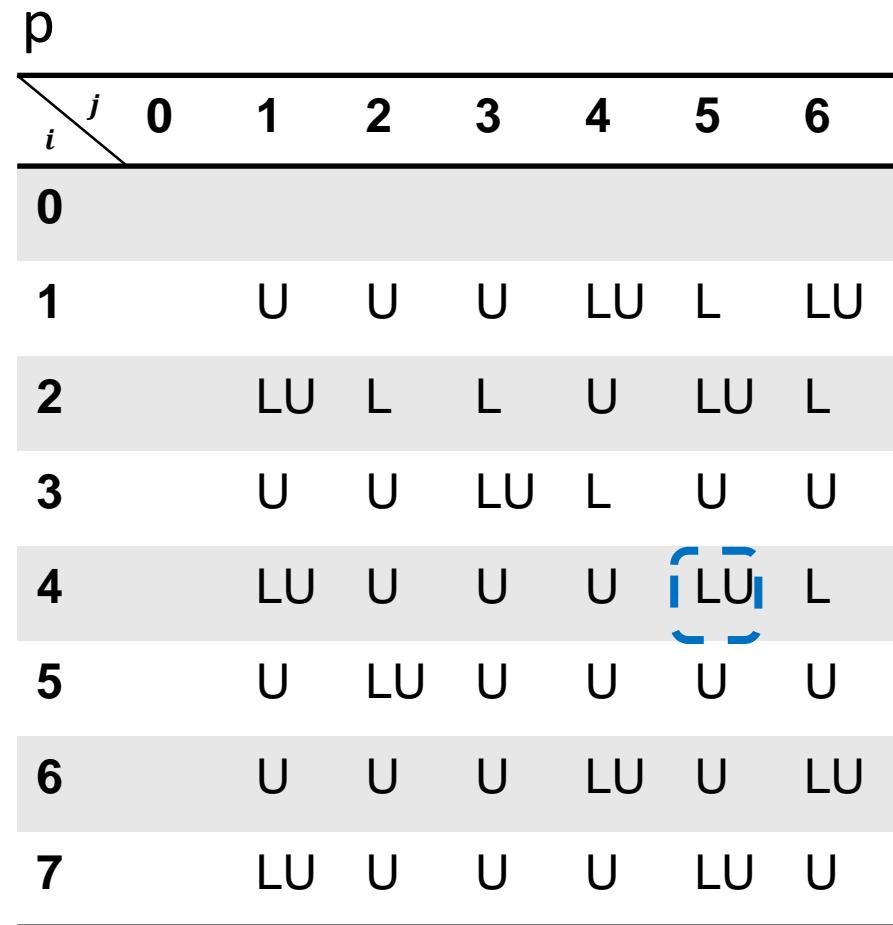
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = $< A >$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = $\langle B, A \rangle$

Example of Optimal Solution Construction

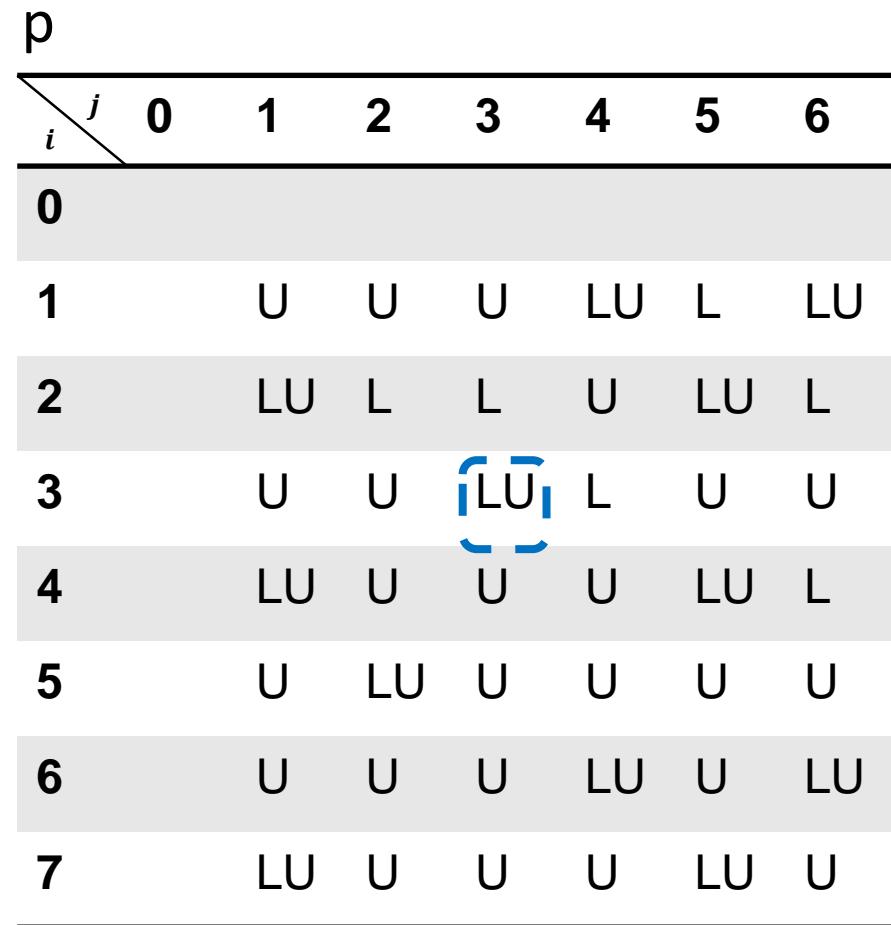
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = $< B, A >$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = < C, B, A >

Example of Optimal Solution Construction

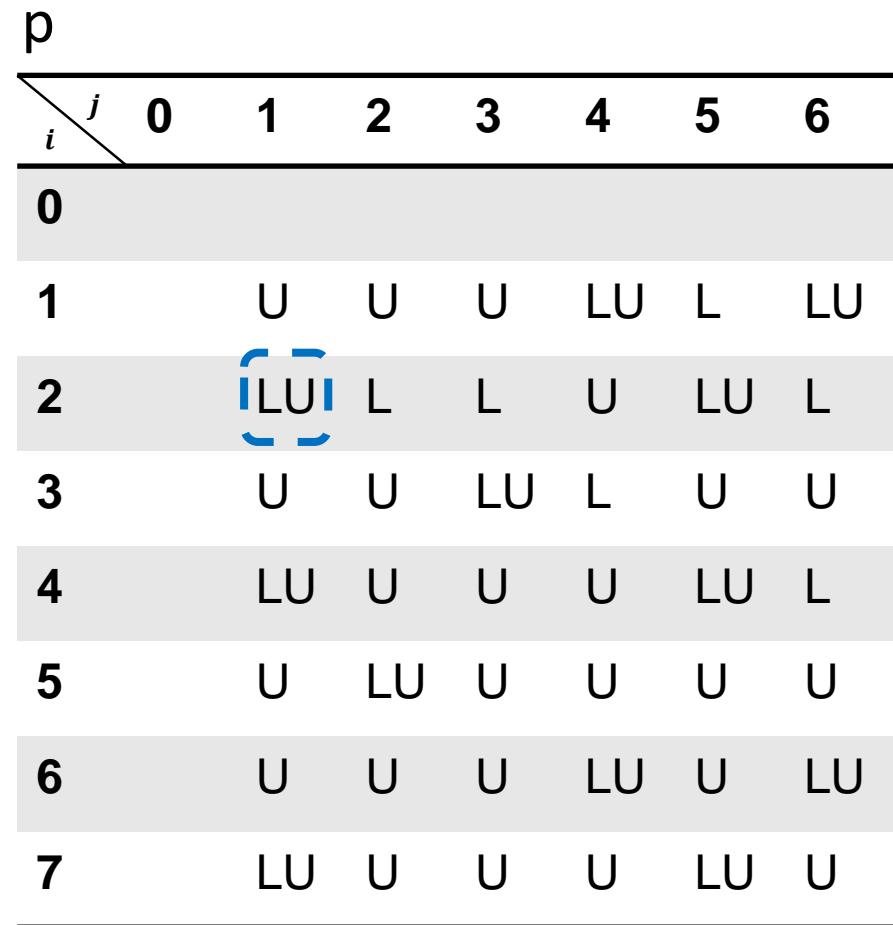
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = $< C, B, A >$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Longest Common
Subsequence = $\langle B, C, B, A \rangle$

Outline

- Review to Part II
- Chain Matrix Multiplication Problem
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Longest common substring

Given two strings $X = x_1x_2\dots x_m$ and $Y = y_1y_2\dots y_n$, we wish to find their longest common substring Z , that is the largest k for which there are indices i and j with $x_i x_{i+1} \dots x_{i+k-1} = y_j y_{j+1} \dots y_{j+k-1}$.

Longest common substring

Given two strings $X = x_1x_2\dots x_m$ and $Y = y_1y_2\dots y_n$, we wish to find their longest common substring Z , that is the largest k for which there are indices i and j with $x_i x_{i+1} \dots x_{i+k-1} = y_j y_{j+1} \dots y_{j+k-1}$.

For example:

X : DEADBEE

Y : EATBEE

Z : BEE //pick the longest contiguous substring

A Dynamic Programming Algorithm

Step 1: Space of Subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d_{i,j}$ to be the length of the longest common substring of $X[1..i]$ and $Y [1..j]$.

A Dynamic Programming Algorithm

Step 1: Space of Subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d_{i,j}$ to be the length of the longest common substring of $X[1..i]$ and $Y [1..j]$.
- In other words, $d_{i,j}$ is the length of the longest common substring ending at x_i and y_j .

A Dynamic Programming Algorithm

Step 1: Space of Subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d_{i,j}$ to be the length of the longest common substring of $X[1..i]$ and $Y [1..j]$.
- In other words, $d_{i,j}$ is the length of the longest common substring ending at x_i and y_j .
- Let D be the mxn matrix $[d_{i,j}]$.

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of X and Y ending at x_{i-1} and y_{j-1}

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of X and Y ending at x_{i-1} and y_{j-1}

Case 2: If $x_i \neq y_j$, then there can't be a common substring ending at x_i and y_j

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of X and Y ending at x_{i-1} and y_{j-1}

Case 2: If $x_i \neq y_j$, then there can't be a common substring ending at x_i and y_j

$$d_{i,j} = \begin{cases} d_{i-1,j-1} + 1, & \text{if } x_i = y_j \\ 0, & \text{if } x_i \neq y_j \end{cases}$$

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

Case 1: If $x_i = y_j$, then $z_k = x_i = y_j$ and Z_{k-1} is a LCS of X and Y ending at x_{i-1} and y_{j-1}

Case 2: If $x_i \neq y_j$, then there can't be a common substring ending at x_i and y_j

$$d_{i,j} = \begin{cases} d_{i-1,j-1} + 1, & \text{if } x_i = y_j \\ 0, & \text{if } x_i \neq y_j \end{cases}$$

Finally, we can have the longest common substring by finding the maximum of what we have computed for all possible ending positions i and j.

$$\text{LCSString}(X, Y) = \max\{d_{i,j}\}$$

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

Again, we do same as the Longest Common Subsequence problem that we initially setup the first row and column of the matrix $d[0, j]$ and $d[i, 0]$ to 0.

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

Again, we do same as the Longest Common Subsequence problem that we initially setup the first row and column of the matrix $d[0, j]$ and $d[i, 0]$ to 0.

Calculate $d[i, j]$ for $j = 1, 2, \dots, n$

Then, the $d[i, j]$ for $i = 1, 2, \dots, m$

So, we fill the matrix table row by row and left to right.

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

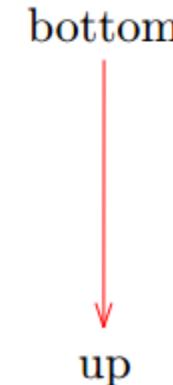
Again, we do same as the Longest Common Subsequence problem that we initially setup the first row and column of the matrix $d[0, j]$ and $d[i, 0]$ to 0.

Calculate $d[i, j]$ for $j = 1, 2, \dots, n$

Then, the $d[i, j]$ for $i = 1, 2, \dots, m$

So, we fill the matrix table row by row and left to right.

$D[i, j]$	$j = 0$	1	2	3	\dots	\dots	n
$i = 0$	0	0	0	0	\dots	\dots	0
1	0						
2	0						
:	0						
m	0						



A Dynamic Programming Algorithm

Step 4: Construction of Optimal Solution

But we **do not need to create another mxn matrix for storing arrows**. Instead, we have I_{\max} and p_{\max} to store the largest length of common substring and its i position respectively. So, we could reconstruct the elements later on.

The Dynamic Programming Algorithm

Longest-Common-Substring(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Longest common substring of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ be a new 2-dimension array;

$l_{max} \leftarrow 0;$

$p_{max} \leftarrow 0;$

The Dynamic Programming Algorithm

Longest-Common-Substring(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Longest common substring of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ be a new 2-dimension array;

$l_{max} \leftarrow 0;$

$p_{max} \leftarrow 0;$

//Initialization

for $i \leftarrow 0$ to m **do**

 | $d[i, 0] \leftarrow 0;$

end

for $j \leftarrow 0$ to n **do**

 | $d[0, j] \leftarrow 0;$

end

The Dynamic Programming Algorithm

```
//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i \neq y_j$  then
            |  $d[i, j] \leftarrow 0$ ;
        end
```

The Dynamic Programming Algorithm

```
//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i \neq y_j$  then
            |  $d[i, j] \leftarrow 0$ ;
        end
        else
            |  $d[i, j] \leftarrow d[i - 1, j - 1] + 1$ ;
            | if  $d[i, j] > l_{max}$  then
                |   |  $l_{max} \leftarrow d[i, j]$ ;
                |   |  $p_{max} \leftarrow i$ ;
            end
        end
    end
end
return  $l_{max}, p_{max}$ ;
```

The Dynamic Programming Algorithm

```
//Dynamic Programming
for  $i \leftarrow 1$  to  $m$  do
    for  $j \leftarrow 1$  to  $n$  do
        if  $x_i \neq y_j$  then
            |  $d[i, j] \leftarrow 0$ ;
        end
        else
            |  $d[i, j] \leftarrow d[i - 1, j - 1] + 1$ ;
            | if  $d[i, j] > l_{max}$  then
                |   |  $l_{max} \leftarrow d[i, j]$ ;
                |   |  $p_{max} \leftarrow i$ ;
            end
        end
    end
end
return  $l_{max}, p_{max}$ ;
```

- The dynamic programming algorithm runs in $O(mn)$ time.

An Algorithm for Constructing Optimal Solutions

Print-LCSubstring(X, l_{max}, p_{max})

Input: String X , l_{max} and p_{max} are generated from
Longest-Common-Substring.

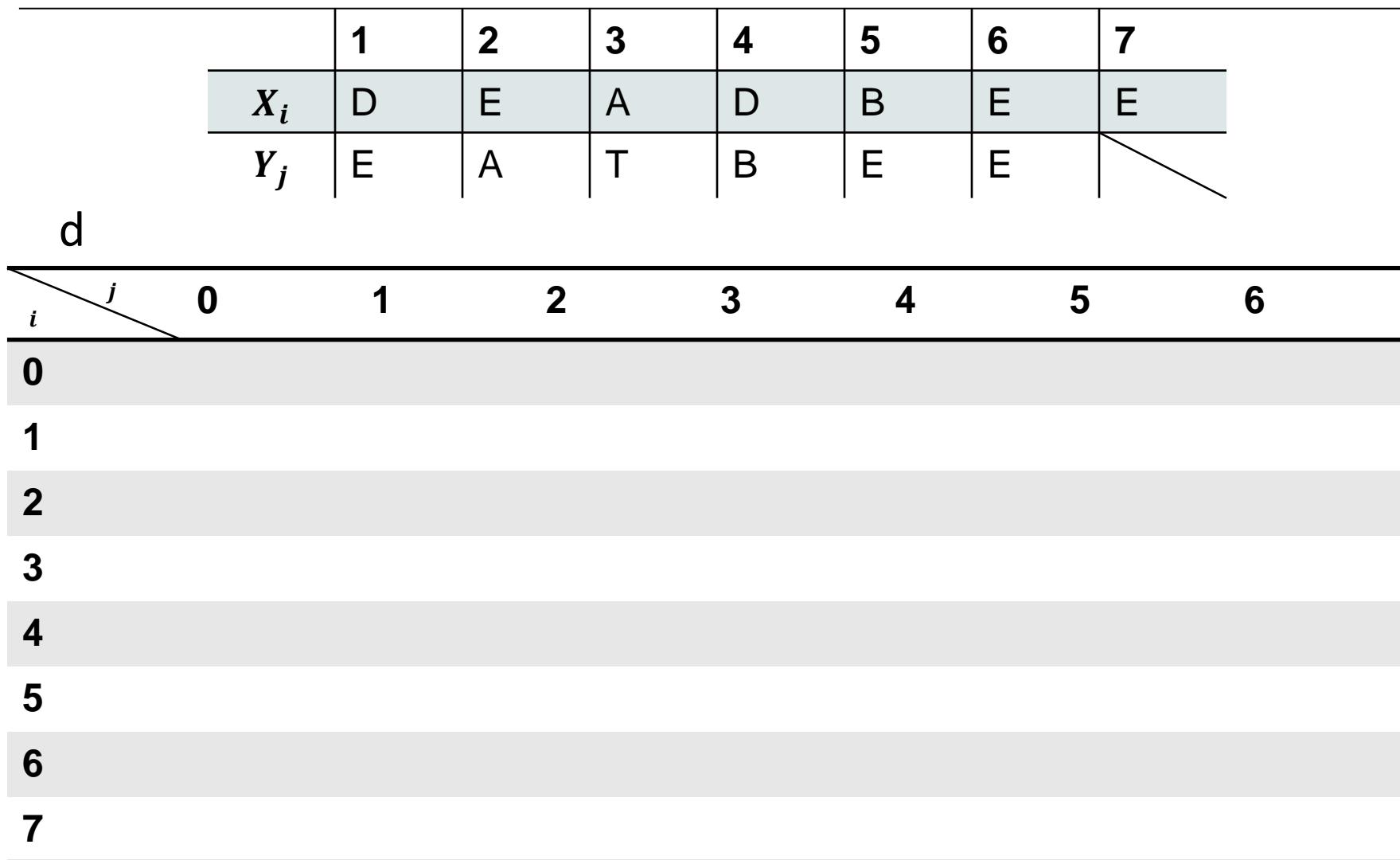
Output: Output the longest common substring of $X[1..i]$ and $Y[1..j]$.

```
if  $l_{max}$  is equal to 0 then
    | return NULL;
end
for  $i \leftarrow (p_{max} - l_{max} + 1)$  to  $p_{max}$  do
    | print  $x_i$ ;
end
```

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

Example of Optimal Solution Construction



$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

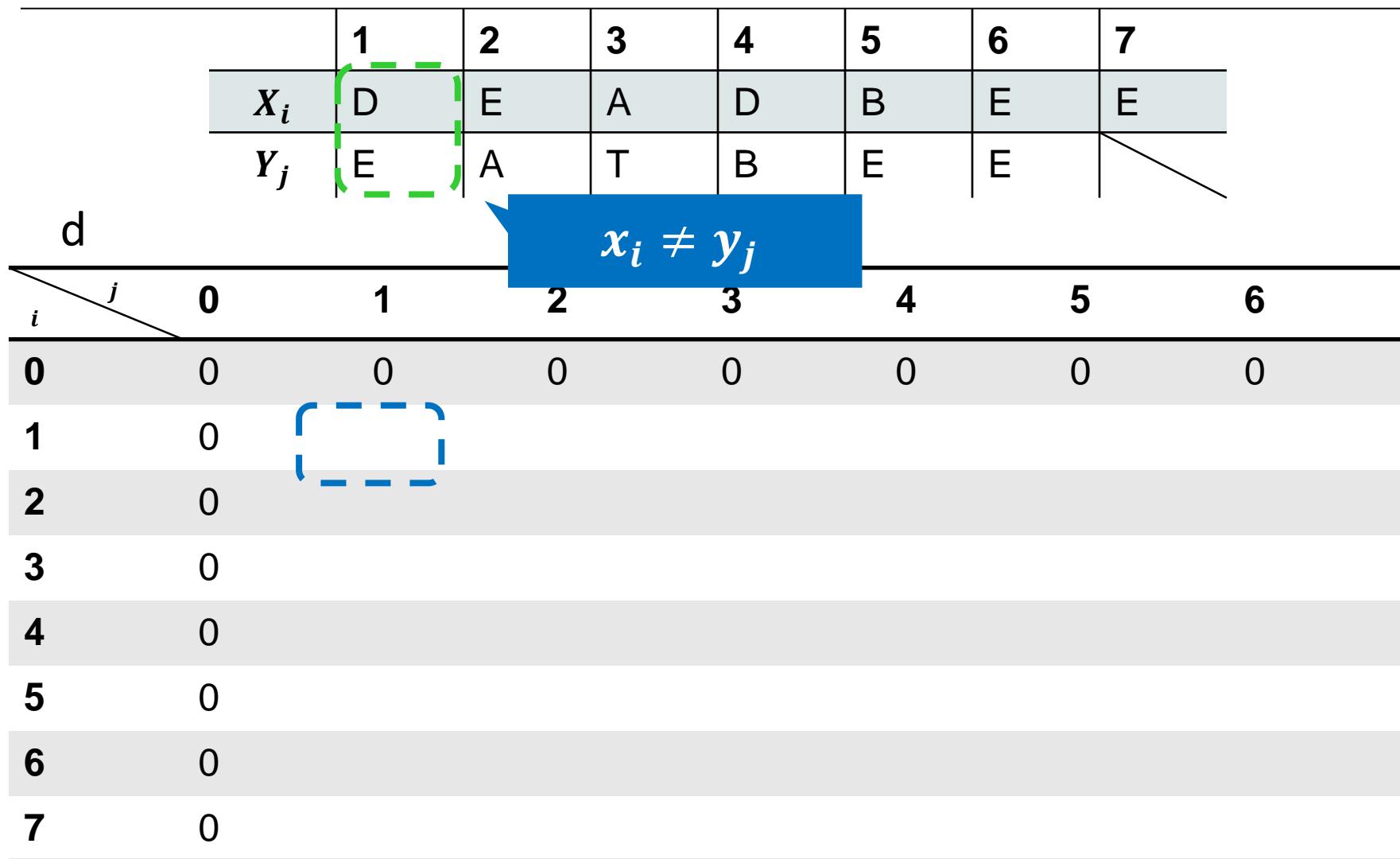
$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0						
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

Initialization

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction



$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	
d							
i \ j	0	1	2	3	4	5	6

0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	
d							
i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0		
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	
d							
i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	
d							
i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	J

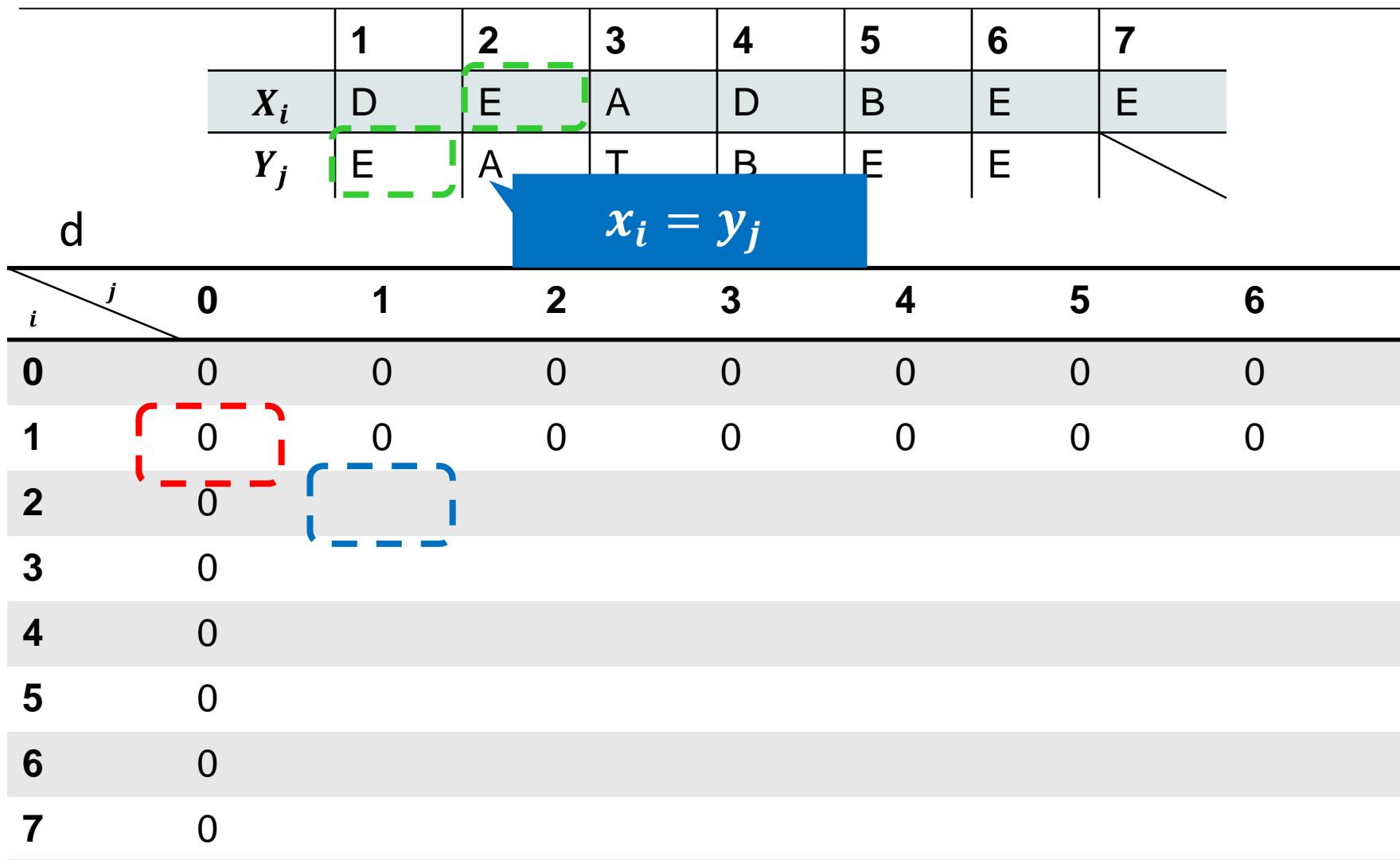
d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0						
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 0$$

$$p_{max} = 0$$

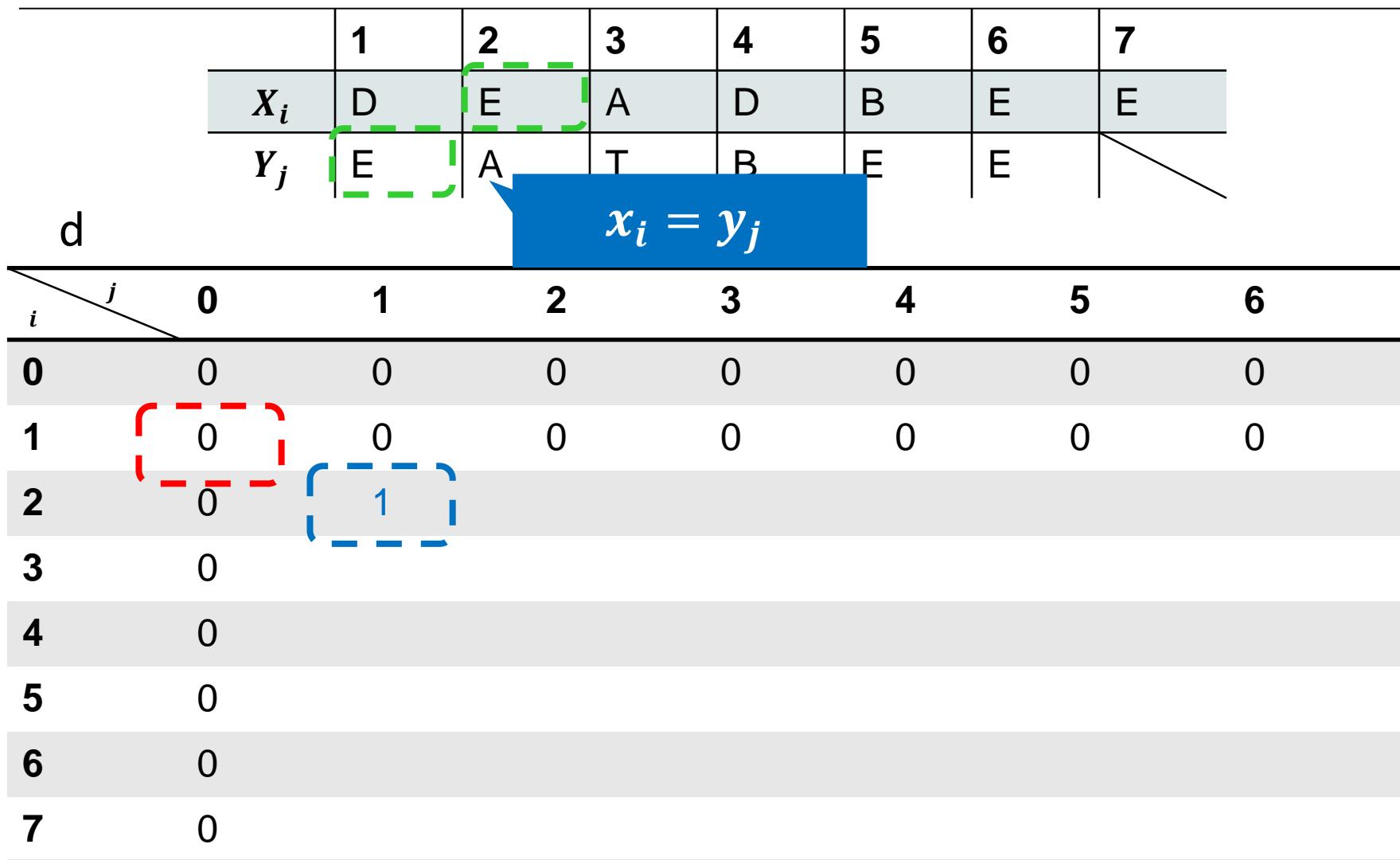
Example of Optimal Solution Construction



$$l_{max} = 0$$

$$p_{max} = 0$$

Example of Optimal Solution Construction



$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0				
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0			
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0		
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	1
3	0						
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0

$$l_{max} = 1$$

$$p_{max} = 2$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2				
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0						
5	0						
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	
d							
i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0						
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0			
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1		
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0						
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1					
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0		
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d	0	1	2	3	4	5	6
i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	1
7	0						

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	1
7	0	1					

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

i \ j	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	1
7	0	1	0	0	0		

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	
Y_j	E	A	T	B	E	E	

d	0	1	2	3	4	5	6
i \ j	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	1
7	0	1	0	0	0	1	

$$l_{max} = 2$$

$$p_{max} = 3$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d	0	1	2	3	4	5	6
i \ j	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0	0	0	2	1
7	0	1	0	0	0	1	3

$$l_{max} = 3$$

$$p_{max} = 7$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d	0	1	2	3	4	5	6
i \ j	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1	0				
7	0	1	0				

Longest Common
Substring

$$l_{max} = 3$$

$$p_{max} = 7$$

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	D	E	A	D	B	E	E
Y_j	E	A	T	B	E	E	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	1	0	0	0	1	1
3	0	0	2	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0
6	0	1					1
7	0	1					3

$$l_{max} = 3$$

$$p_{max} = 7$$

$$LCSubstring = BEE$$

Outline

- Review to Part II
- Chain Matrix Multiplication Problem
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Motivations and Applications

- Spell Correction:
 - The user typed “graffe”
 - Which is the closest?
 - graf
 - graft
 - grail
 - Giraffe

Motivations and Applications

- Spell Correction:
 - The user typed “graffe”
 - Which is the closest?
 - graf
 - graft
 - grail
 - Giraffe
 - When you mis-type "algorithm" as "alogrthm" at Google, you may be delighted that the search engine has corrected the spelling error for you

Motivations and Applications

- Computational Biology:
 - Scientists need to align two sequences of DNA

AGGCTATCACCTGACCTCCAGGCCGATGCC

TAGCTATCACGACCGCGGTGATTGCCCGAC

Motivations and Applications

- Computational Biology:
 - Scientists need to align two sequences of DNA

AGGCTATCACCTGACCTCCAGGCCGATGCC

TAGCTATCACGACC CGCGGTGATTGCCCGAC

- Resulting alignment:

-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---

TAG-CTATCAC--GACC GC--GGTCGATTGCCCGAC

Motivations and Applications

- Computational Biology:
 - Scientists need to align two sequences of DNA

AGGCTATCACCTGACCTCCAGGCCGATGCC

TAGCTATCACGACC CGCGGTGATTGCCCGAC

- Resulting alignment:

-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---

TAG-CTATCAC--GACC GC--GGTCGATTGCCCGAC

- Also for Machine Translation, Information Extraction, Speech Recognition, etc.

Outline

- Review to Part II
- Chain Matrix Multiplication Problem
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:
 - ① delete a -> bode

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:
 - ① delete a -> bode
 - ② insert l after b -> blode

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:
 - ① delete a -> bode
 - ② insert l after b -> blode
 - ③ delete d -> bloe

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:
 - ① delete a -> bode
 - ② insert l after b -> blode
 - ③ delete d -> bloe
 - ④ substitute e with g -> blog

Minimum Edit Distance

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, the edit distance is the **smallest number** of following **edit operations** to turn X into Y :
 - Insertion: add a letter
 - Deletion: remove a letter
 - Substitution: replace a character with another one.
- Example:
 - Consider that $X = \text{abode}$ and $Y = \text{blog}$. The edit distance is 4. We can change abode into blog by 4 operations:
 - ① delete a -> bode
 - ② insert l after b -> blode
 - ③ delete d -> bloe
 - ④ substitute e with g -> blog
 - Impossible** to do so with **3** operations!

Minimum Edit Distance

- Two strings and their alignment:

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N

Minimum Edit Distance

- Two strings and their alignment:

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N

- If each operation has cost of 1
 - Distance between these is 5

Minimum Edit Distance

- Two strings and their alignment:

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N

- If each operation has cost of 1
 - Distance between these is 5
- If substitutions cost 2
 - Distance between them is 8

Outline

- Review to Part II
- Chain Matrix Multiplication Problem
 - Review of Matrix Multiplication
 - The Chain Matrix Multiplication Problem
 - A Dynamic Programming Algorithm
- Longest Common Subsequence Problem
 - Longest Common Subsequence Problem
 - Longest Common Substring Problem
- Minimum Edit Distance Problem
 - Motivations and Applications
 - Minimum Edit Distance Problem
 - A Dynamic Programming Algorithm

Developing A Dynamic Programming Algorithm

- Given two strings $X=(x_1, x_2, \dots, x_m)$ and $Y=(y_1, y_2, \dots, y_n)$, we want to find the minimum edit distance and the corresponding sequence of operations.

Developing A Dynamic Programming Algorithm

Step 1: Space of subproblems

For $1 \leq i \leq m$, and $1 \leq j \leq n$,

- Define $d[i,j]$ to be the **minimum edit distance** of the substrings $X[1..i]$ and $Y [1..j]$.

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Case 1: Turn $X[1..i-1]$ into $Y[1..j]$ and delete $X[i]$

$$\text{Example 1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{dab}) + 1$$

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Case 1: Turn $X[1..i-1]$ into $Y[1..j]$ and delete $X[i]$

$$\text{Example 1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{dab}) + 1$$

Case 2: Turn $X[1..i]$ into $Y[1..j-1]$ and insert $Y[j]$

$$\text{Example 2 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cxy} \rightarrow \text{da}) + 1$$

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Case 1: Turn $X[1..i-1]$ into $Y[1..j]$ and delete $X[i]$

$$\text{Example 1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{dab}) + 1$$

Case 2: Turn $X[1..i]$ into $Y[1..j-1]$ and insert $Y[j]$

$$\text{Example 2 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cxy} \rightarrow \text{da}) + 1$$

Case 3: Turn $X[1..i-1]$ into $Y[1..j-1]$ and substitute $X[i]$ with $Y[j]$ if needed ($X[i] \neq Y[j]$)

$$\text{Example 3.1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{da}) + 1$$

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Case 1: Turn $X[1..i-1]$ into $Y[1..j]$ and delete $X[i]$

$$\text{Example 1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{dab}) + 1$$

Case 2: Turn $X[1..i]$ into $Y[1..j-1]$ and insert $Y[j]$

$$\text{Example 2 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cxy} \rightarrow \text{da}) + 1$$

Case 3: Turn $X[1..i-1]$ into $Y[1..j-1]$ and substitute $X[i]$ with $Y[j]$ if needed ($X[i] \neq Y[j]$)

$$\text{Example 3.1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{da}) + 1$$

$$\text{Example 3.2 : } \text{MED}(\text{cxy} \rightarrow \text{day}) = \text{MED}(\text{cx} \rightarrow \text{da})$$

Relationships among Subproblems

Step 2: Relating the value of a problem and those of its subproblems

To turn $X[1..i]$ into $Y[1..j]$, we have three cases:

Case 1: Turn $X[1..i-1]$ into $Y[1..j]$ and delete $X[i]$

$$\text{Example 1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{dab}) + 1$$

Case 2: Turn $X[1..i]$ into $Y[1..j-1]$ and insert $Y[j]$

$$\text{Example 2 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cxy} \rightarrow \text{da}) + 1$$

Case 3: Turn $X[1..i-1]$ into $Y[1..j-1]$ and substitute $X[i]$ with $Y[j]$ if needed ($X[i] \neq Y[j]$)

$$\text{Example 3.1 : } \text{MED}(\text{cxy} \rightarrow \text{dab}) = \text{MED}(\text{cx} \rightarrow \text{da}) + 1$$

$$\text{Example 3.2 : } \text{MED}(\text{cxy} \rightarrow \text{day}) = \text{MED}(\text{cx} \rightarrow \text{da})$$

$$D[i, j] = \min \left\{ \begin{array}{l} D[i - 1, j] + 1 \\ D[i, j - 1] + 1 \\ D[i - 1, j - 1] + \begin{cases} 0 & \text{if } X[i] = Y[j] \\ 1 & \text{otherwise.} \end{cases} \end{array} \right.$$

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

- Initially, we fill the first row of the matrix $D[0,j]$ with j and the first column $D[i, 0]$ with i .

$D[i,j]$	$j=0$	1	2	3	...	n
$i=0$	0	1	2	3	...	n
1	1	→				
2	2	→				
...	...	→				
m	m	→				

Bottom
↓
Up

A Dynamic Programming Algorithm

Step 3: Bottom-up Computation

- Initially, we fill the first row of the matrix $D[0,j]$ with j and the first column $D[i, 0]$ with i .
- Process the other terms in the recursive manner.

$D[i,j]$	$j=0$	1	2	3	...	n
$i=0$	0	1	2	3	...	n
1	1					
2	2					
...	...					
m	m					

Bottom
↓
Up

A Dynamic Programming Algorithm

Step 4: Construction of Optimal Solution

- Also, we create another matrix $p[i, j]$ ($1 \leq i \leq m, 1 \leq j \leq n$) to store arrows pointing to the element that was used in the computation, which is used to restore the optimal sequence of the operations.

A Dynamic Programming Algorithm

Step 4: Construction of Optimal Solution

- Also, we create another matrix $p[i, j]$ ($1 \leq i \leq m, 1 \leq j \leq n$) to store arrows pointing to the element that was used in the computation, which is used to restore the optimal sequence of the operations.

$$D[i, j] = \min \begin{cases} D[i - 1, j] + 1 & \text{Deletion} \\ D[i, j - 1] + 1 & \text{Insertion} \\ D[i - 1, j - 1] + \begin{cases} 0 & \text{if } X[i] = Y[j] \\ 1 & \text{otherwise.} \end{cases} & \text{Substitution} \end{cases}$$

$$p[i, j] = \begin{cases} \textit{Left} & \text{Insertion} \\ \textit{Up} & \text{Deletion} \\ \textit{Left up} & \text{Substitution} \end{cases}$$

The Dynamic Programming Algorithm

Minimum-Edit-Distance(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Minimum edit distance of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ and $p[0..m, 0..n]$ be two new 2-dimension arrays;

The Dynamic Programming Algorithm

Minimum-Edit-Distance(X, Y)

Input: Two strings \mathbf{X}, \mathbf{Y} .

Output: Minimum edit distance of X and Y .

$m \leftarrow \text{length}(X);$

$n \leftarrow \text{length}(Y);$

Let $d[0..m, 0..n]$ and $p[0..m, 0..n]$ be two new 2-dimension arrays;

//Initialization

for $i \leftarrow 0$ to m **do**

$d[i, 0] \leftarrow i;$

$p[i, 0] \leftarrow "U";$

end

for $j \leftarrow 0$ to n **do**

$d[0, j] \leftarrow j;$

$p[0, j] \leftarrow "L";$

end

The Dynamic Programming Algorithm

//Dynamic Programming

for $i \leftarrow 1$ to m do

 for $j \leftarrow 1$ to n do

 if x_i is not equal to y_j then

 | $c \leftarrow 1$;

 end

 else

 | $c \leftarrow 0$;

 end

$$D[i, j] = \min \begin{cases} D[i - 1, j] + 1 \\ D[i, j - 1] + 1 \\ D[i - 1, j - 1] + \begin{cases} 0 & \text{if } X[i] = Y[j] \\ 1 & \text{otherwise.} \end{cases} \end{cases}$$

The Dynamic Programming Algorithm

```
//Dynamic Programming
for i ← 1 to m do
    for j ← 1 to n do
        if  $x_i$  is not equal to  $y_j$  then
            | c ← 1;
        end
        else
            | c ← 0;
        end
        if  $d[i - 1, j - 1] + c \leq d[i - 1][j] + 1$  and
            $d[i - 1, j - 1] + c \leq d[i][j - 1] + 1$  then
            |  $d[i, j] \leftarrow d[i - 1, j - 1] + c;$ 
            |  $p[i, j] \leftarrow "LU"; // "LU" indicates left up arrow.$ 
        end
```

The Dynamic Programming Algorithm

```

//Dynamic Programming
for i ← 1 to m do
    for j ← 1 to n do
        if  $x_i$  is not equal to  $y_j$  then
            | c ← 1;
        end
        else
            | c ← 0;
        end
        if  $d[i - 1, j - 1] + c \leq d[i - 1][j] + 1$  and
            $d[i - 1, j - 1] + c \leq d[i][j - 1] + 1$  then
            |  $d[i, j] \leftarrow d[i - 1, j - 1] + c;$ 
            |  $p[i, j] \leftarrow "LU"; // "LU" indicates left up arrow.$ 
        end
        else if  $d[i, j - 1] + 1 < d[i - 1][j] + 1$  and
                $d[i, j - 1] + 1 < d[i - 1, j - 1] + c$  then
            |  $d[i, j] \leftarrow d[i, j - 1] + 1;$ 
            |  $p[i, j] \leftarrow "L"; // "L" indicates up arrow.$ 
        end
        else
            |  $d[i, j] \leftarrow d[i - 1, j] + 1;$ 
            |  $p[i, j] \leftarrow "U"; // "U" indicates left arrow.$ 
        end
    end
end
return d, p;

```

The Dynamic Programming Algorithm

```

//Dynamic Programming
for i ← 1 to m do
    for j ← 1 to n do
        if  $x_i$  is not equal to  $y_j$  then
            | c ← 1;
        end
        else
            | c ← 0;
        end
        if  $d[i - 1, j - 1] + c \leq d[i - 1][j] + 1$  and
            $d[i - 1, j - 1] + c \leq d[i][j - 1] + 1$  then
            |  $d[i, j] \leftarrow d[i - 1, j - 1] + c;$ 
            |  $p[i, j] \leftarrow "LU"; // "LU" indicates left up arrow.$ 
        end
        else if  $d[i, j - 1] + 1 < d[i - 1][j] + 1$  and
                $d[i, j - 1] + 1 < d[i - 1, j - 1] + c$  then
            |  $d[i, j] \leftarrow d[i, j - 1] + 1;$ 
            |  $p[i, j] \leftarrow "L"; // "L" indicates up arrow.$ 
        end
        else
            |  $d[i, j] \leftarrow d[i - 1, j] + 1;$ 
            |  $p[i, j] \leftarrow "U"; // "U" indicates left arrow.$ 
        end
    end
end
return d, p;

```

Time Cost: O(mn)

An Algorithm for Constructing Optimal Solutions

Algorithm 3: Print-MED(p, X, i, j)

Input: Array p generated from Minimum-Edit-Distance, string X , index i and j .

Output: Output the sequence of operations.

if i is equal to 0 and j is equal to 0 **then**

 | **return** NULL;

end

if $p[i, j]$ is equal to "LU" **then**

 | Print-MED($p, X, i - 1, j - 1$);

 | **if** x_i is equal to y_j **then**

 | print "Do nothing"

 | **end**

 | **else**

 | print "Substitute x_i with y_j ;"

 | **end**

end

An Algorithm for Constructing Optimal Solutions

Algorithm 3: Print-MED(p, X, i, j)

Input: Array p generated from Minimum-Edit-Distance, string X , index i and j .

Output: Output the sequence of operations.

if i is equal to 0 and j is equal to 0 **then**

| **return** NULL;

end

if $p[i, j]$ is equal to "LU" **then**

| Print-MED($p, X, i - 1, j - 1$);

| **if** x_i is equal to y_j **then**

| | print "Do nothing"

| **end**

| **else**

| | print "Substitute x_i with y_j ;"

| **end**

end

else if $p[i, j]$ is equal to "U" **then**

| Print-MED($p, X, i - 1, j$);

| print "Delete x_i ";

end

else

| Print-MED($p, X, i, j - 1$);

| print "Insert y_j "

end

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d



0

1

2

3

4

5

6

7

p



0

1

2

3

4

5

6

7

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1 1

2 2

3 3

4 4

5 5

6 6

7 7

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1 U

2 U

3 U

4 U

5 U

6 U

7 U

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

p

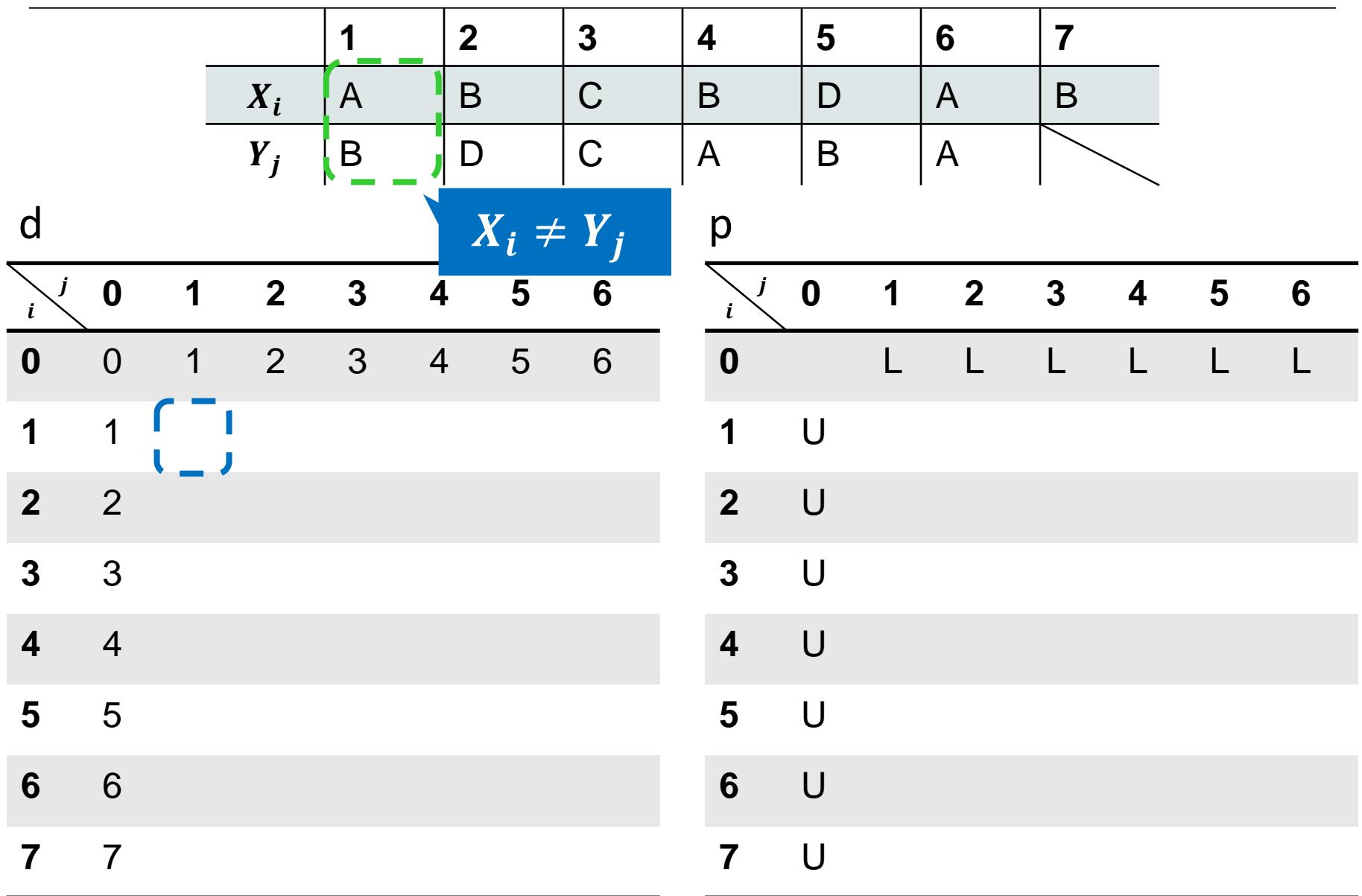
$i \backslash j$	0	1	2	3	4	5	6
0	0	L	L	L	L	L	L

Initialization

1	1
2	2
3	3
4	4
5	5
6	6
7	7

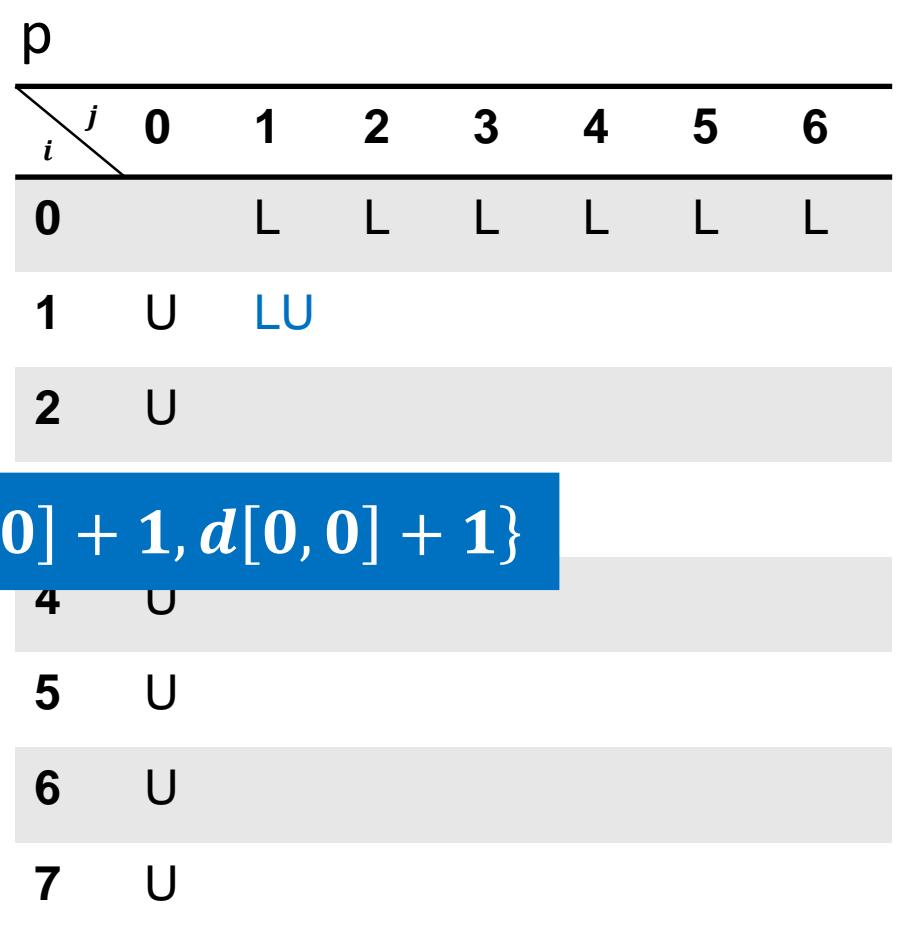
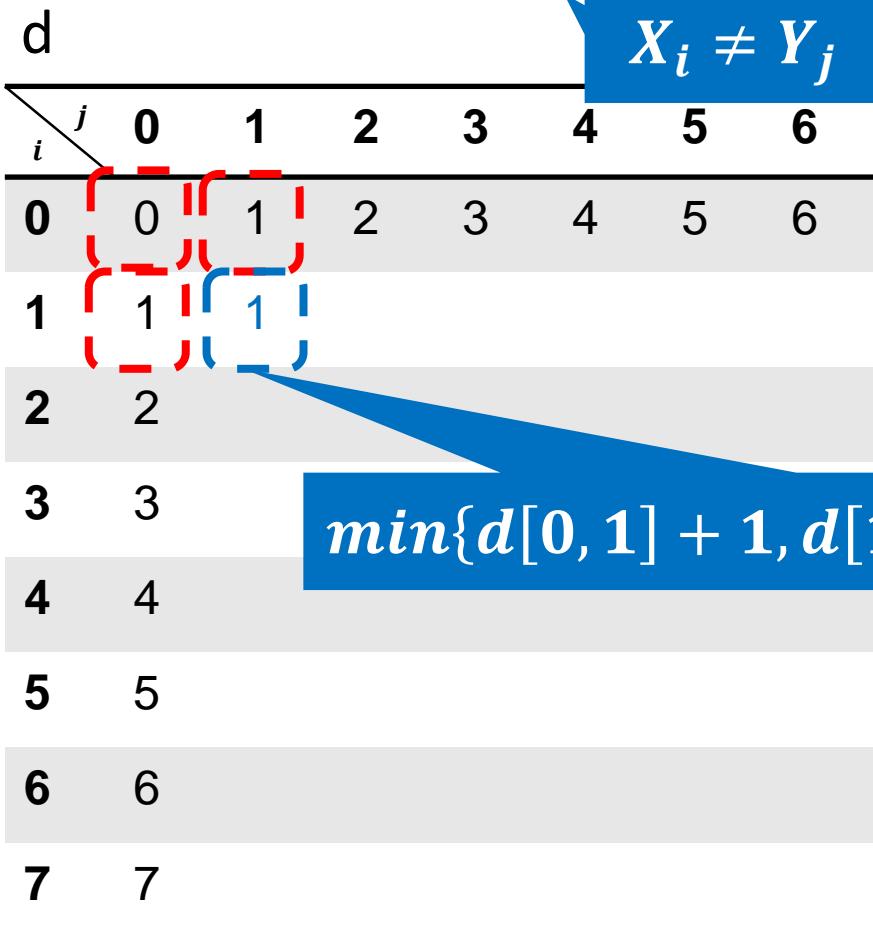
2	U
3	U
4	U
5	U
6	U
7	U

Example of Optimal Solution Construction



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2				
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU				
2	U						
3	U						
4	U						
5	U						
6	U						
7	U						

2	2
3	3
4	4
5	5
6	6
7	7

2	U
3	U
4	U
5	U
6	U
7	U

Example of Optimal Solution Construction

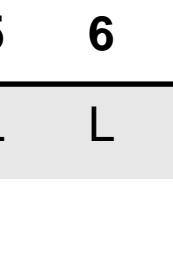
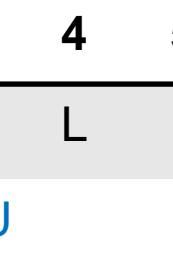
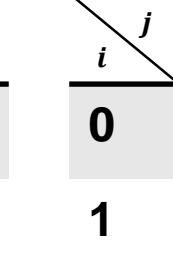
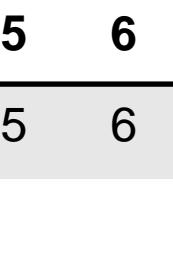
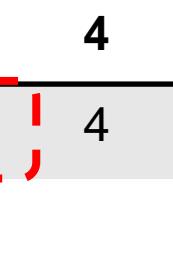
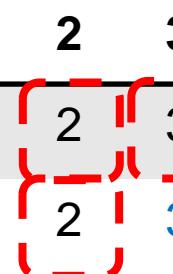
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3			
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU		
2	U						
3	U						
4	U						
5	U						
6	U						
7	U						



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	
d							
$i \backslash j$	0	1	2	3	4	5	6

	0	1	2	3	4	5	6
$i \backslash j$	0	0	1	2	3	4	5
0	0	1	2	3	4	5	6

	0	1	2	3	4	5	6
$i \backslash j$	0	L	L	L	L	L	L
1	U	U	LU	LU	LU	LU	

2	2	$d[i - 1, j - 1] + 0$					
3	3						
4	4						
5	5						
6	6						
7	7						

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	LU	L
2	U						
3	U						
4	U						
5	U						
6	U						
7	U						

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U						
3	U						
4	U						
5	U						
6	U						
7	U						

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1					
---	---	---	--	--	--	--	--

3	3						
---	---	--	--	--	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU					
---	---	----	--	--	--	--	--

3	U						
---	---	--	--	--	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

	j	0	1	2	3	4	5	6
i	0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2				
---	---	---	---	--	--	--	--

3	3						
---	---	--	--	--	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

	j	0	1	2	3	4	5	6
i	0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU				
---	---	----	----	--	--	--	--

3	U						
---	---	--	--	--	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3			
---	---	---	---	---	--	--	--

3	3						
---	---	--	--	--	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU		
---	---	----	----	----	----	--	--

3	U						
---	---	--	--	--	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
2	2	1	2	3	3	4	5

2	2	1	2	3	3	4	
3	3						

3	3						
4	4						

5	5						
6	6						

7	7						
8	8						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	

3	U						
4	U						

5	U						
6	U						

6	U						
7	U						

7	U						
8	U						

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
2	2	1	2	3	4	4	3
3	3						
4	4						
5	5						
6	6						
7	7						

3	U
4	U
5	U
6	U
7	U

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	LU
3	U						
4	U						
5	U						
6	U						
7	U						

3	U
4	U
5	U
6	U
7	U

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2	1	2	3	4	3	4
3	3						
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U						
4	U						
5	U						
6	U						
7	U						



3 U

4 U

5 U

6 U

7 U

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2					
---	---	---	--	--	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U					
---	---	---	--	--	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2				
---	---	---	---	--	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

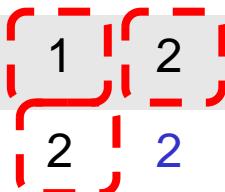
3	U	U	LU				
---	---	---	----	--	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--



1
2

1
2

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2			
---	---	---	---	---	--	--	--

4	4						
---	---	--	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

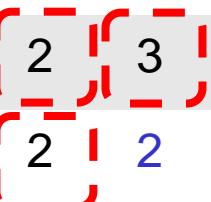
3	U	U	LU	LU			
---	---	---	----	----	--	--	--

4	U						
---	---	--	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--



Example of Optimal Solution Construction

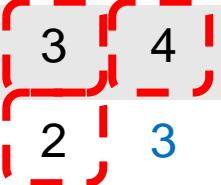
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2	1	2	3	4	3	4
3	3	2	2	2	3		
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L		
4	U						
5	U						
6	U						
7	U						



Example of Optimal Solution Construction

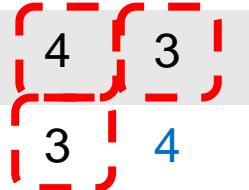
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2	1	2	3	4	3	4
3	3	2	2	2	3	4	
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	
4	U						
5	U						
6	U						
7	U						



Example of Optimal Solution Construction

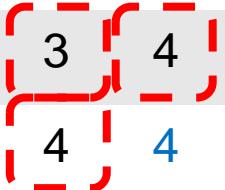
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2	1	2	3	4	3	4
3	3	2	2	2	3	4	4
4	4						
5	5						
6	6						
7	7						

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU
4	U						
5	U						
6	U						
7	U						



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3					
---	---	---	--	--	--	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU					
---	---	----	--	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	3
---	---	---	---	---	---	---	---

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU				
---	---	----	----	--	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3		
---	---	---	---	---	---	--	--

5	5						
---	---	--	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

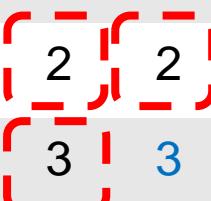
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU			
---	---	----	----	----	--	--	--

5	U						
---	---	--	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	3	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	3
---	---	---	---	---	---	---	---

5	5
---	---

6	6
---	---

7	7
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

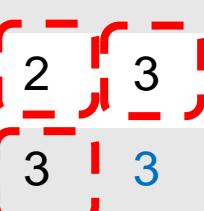
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	LU
---	---	----	----	----	----	----	----

5	U
---	---

6	U
---	---

7	U
---	---



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	3
---	---	---	---	---	---	---	---

5	5
---	---

6	6
---	---

7	7
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

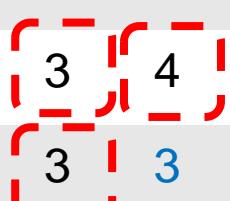
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	LU
---	---	----	----	----	----	----	----

5	U
---	---

6	U
---	---

7	U
---	---



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5
---	---

6	6
---	---

7	7
---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

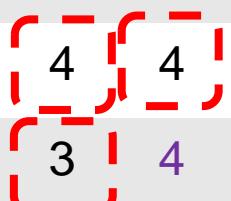
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U
---	---

6	U
---	---

7	U
---	---



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4					
---	---	---	--	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U					
---	---	---	--	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3				
---	---	---	---	--	--	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

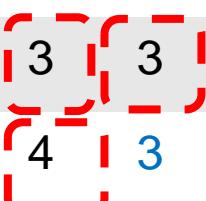
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU				
---	---	---	----	--	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

 X_i Y_j

2

D

3

C

4

B

5

D

6

A

7

B

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	3	4		
---	---	---	---	---	---	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

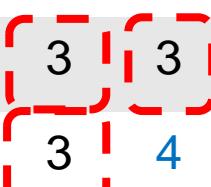
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU			
---	---	---	----	----	--	--	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--



3	3
---	---

3	4
---	---

3	3
---	---

3	4
---	---

3	4
---	---

3	4
---	---

3	4
---	---

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4		
---	---	---	---	---	---	--	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

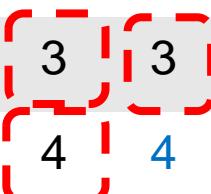
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU		LU	
---	---	---	----	----	--	----	--

6	U						
---	---	--	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--



4

4

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	
---	---	---	---	---	---	---	--

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1		LU	LU	LU	LU	L	LU
---	--	----	----	----	----	---	----

2		LU	LU	LU	LU	LU	L
---	--	----	----	----	----	----	---

3		U	LU	LU	L	L	LU
---	--	---	----	----	---	---	----

4		LU	LU	LU	LU	LU	L
---	--	----	----	----	----	----	---

5		U	LU	LU	LU		LU
---	--	---	----	----	----	--	----

6							
---	--	--	--	--	--	--	--

7							
---	--	--	--	--	--	--	--



3

4

4

4

4

3

3

3

3

3

3

4

4

4

4

4

4

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6						
---	---	--	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

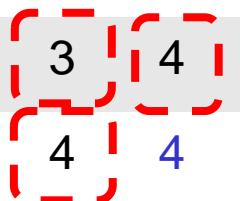
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----	----

6	U							
---	---	--	--	--	--	--	--	--

7	U							
---	---	--	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5					
---	---	---	--	--	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U					
---	---	---	--	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	4	3	4	4	4
---	---	---	---	---	---	---	---

6	6	5	5	4			
---	---	---	---	---	--	--	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U				
---	---	---	---	--	--	--	--

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1 1 1 2 3 3 4 5

2 2 1 2 3 4 3 4

3 3 2 2 2 3 4 4

4 4 3 3 3 3 3 4

5 5 4 3 4 4 4 4

6 6 5 4 4 4

7 7

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1 U LU LU LU LU L LU

2 U LU LU LU LU LU L

3 U U LU LU L L LU

4 U LU LU LU LU LU L

5 U U LU LU LU LU LU

6 U U U LU

7 U

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	4	
---	---	---	---	---	---	---	--

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

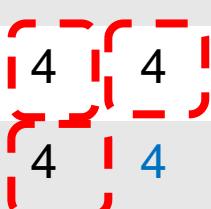
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U						
---	---	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	4	5
---	---	---	---	---	---	---	---

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U						
---	---	--	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7						
---	---	--	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

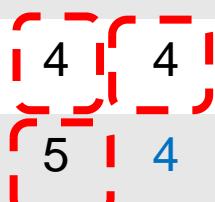
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U						
---	---	--	--	--	--	--	--



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6					
---	---	---	--	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U	LU					
---	---	----	--	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6	5				
---	---	---	---	--	--	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U	LU	U				
---	---	----	---	--	--	--	--

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6	5	5	5		
---	---	---	---	---	---	--	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

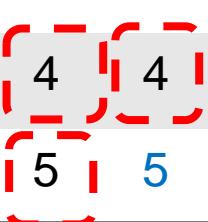
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U	LU	U	LU	LU	LU	LU
---	---	----	---	----	----	----	----



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6	5	5	5	5	
---	---	---	---	---	---	---	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

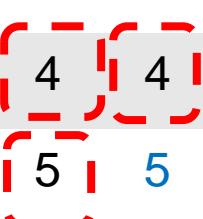
3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U	LU	U	LU	LU	LU	LU
---	---	----	---	----	----	----	----



Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6	5	5	5	4	
---	---	---	---	---	---	---	--

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	U	LU	U	LU	LU	LU	LU
---	---	----	---	----	----	----	----



Example of Optimal Solution Construction

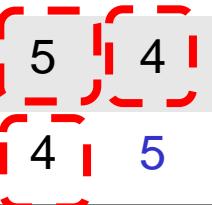
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3	4	5
2	2	1	2	3	4	3	4
3	3	2	2	2	3	4	4
4	4	3	3	3	3	3	4
5	5	4	3	4	4	4	4
6	6	5	4	4	4	5	4
7	7	6	5	5	5	4	5

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU
4	U	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	L



4
5

4
5

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	
Y_j	B	D	C	A	B	A	B

d

$i \backslash j$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6

1	1	1	2	3	3	4	5
---	---	---	---	---	---	---	---

2	2	1	2	3	4	3	4
---	---	---	---	---	---	---	---

3	3	2	2	2	3	4	4
---	---	---	---	---	---	---	---

4	4	3	3	3	3	3	4
---	---	---	---	---	---	---	---

5	5	4	3	4	4	4	4
---	---	---	---	---	---	---	---

6	6	5	4	4	4	5	4
---	---	---	---	---	---	---	---

7	7	6	5	5	5	4	5
---	---	---	---	---	---	---	---

p

$i \backslash j$	0	1	2	3	4	5	6
0	L	L	L	L	L	L	L

1	U	LU	LU	LU	LU	L	LU
---	---	----	----	----	----	---	----

2	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

3	U	U	LU	LU	L	L	LU
---	---	---	----	----	---	---	----

4	U	LU	LU	LU	LU	LU	L
---	---	----	----	----	----	----	---

5	U	U	LU	LU	LU	LU	LU
---	---	---	----	----	----	----	----

6	U	U	U	LU	LU	LU	LU
---	---	---	---	----	----	----	----

7	LU	L					
---	----	---	--	--	--	--	--



Minimum Edit Distance

Example of Optimal Solution Construction

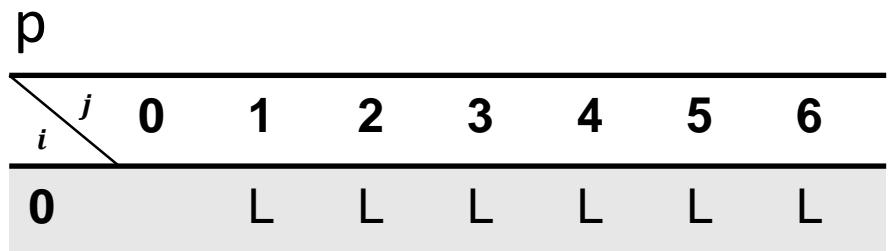
	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

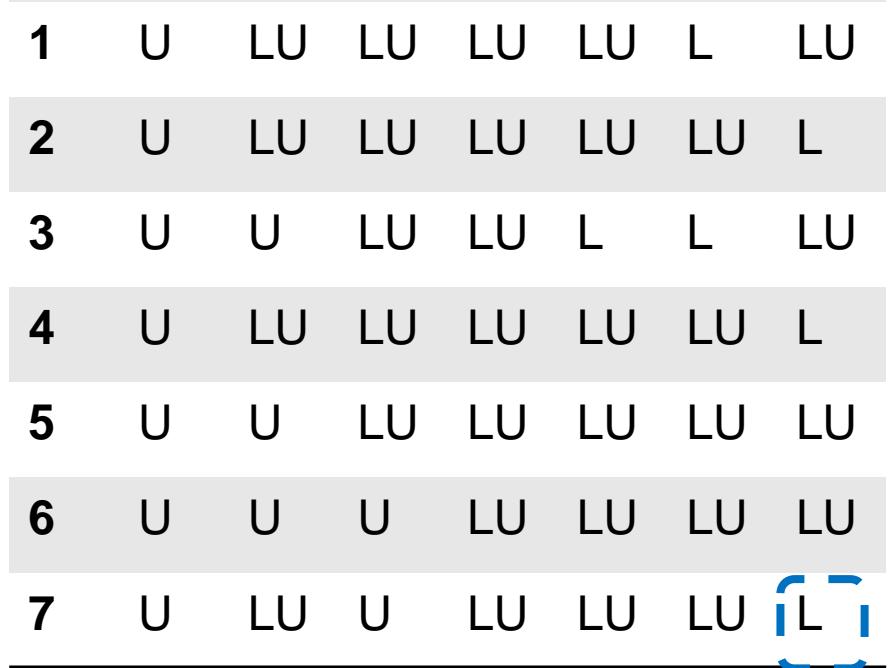
	$i \backslash j$	0	1	2	3	4	5	6
0		L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU	
2	U	LU	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



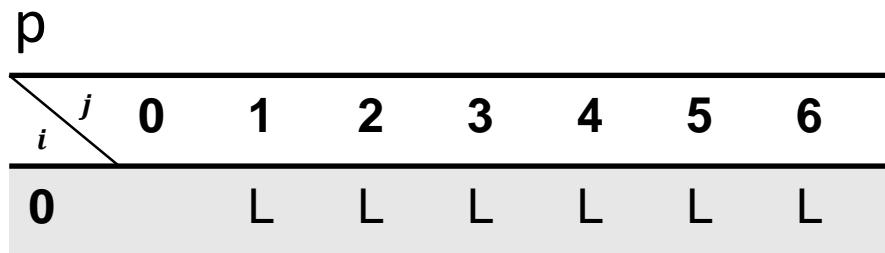
Operations:



Insert A

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Operations:

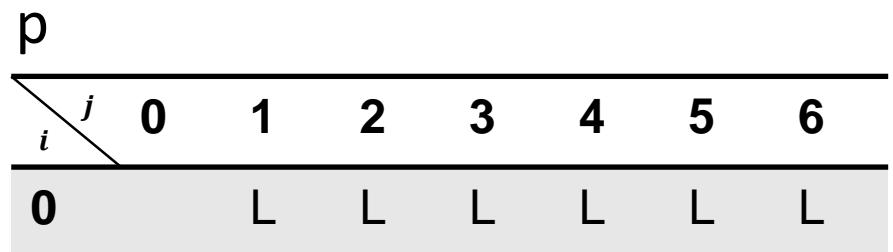
0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU
4	U	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU

Do nothing

Insert A

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	



Operations:

Do nothing

Do nothing

Insert A

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

	$i \backslash j$	0	1	2	3	4	5	6
0		L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU	
2	U	LU	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

Substitute D with C

Do nothing

Do nothing

Insert A

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

Substitute B with D

Substitute D with C

Do nothing

Do nothing

Insert A

	j	0	1	2	3	4	5	6
i	0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU	
2	U	LU	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

Delete C

Substitute B with D

Substitute D with C

Do nothing

Do nothing

Insert A

	p	0	1	2	3	4	5	6
i \ j	0	L	L	L	L	L	L	L
0	U	LU	LU	LU	LU	L	LU	
1	U	LU	LU	LU	LU	LU	L	LU
2	U	LU	LU	LU	LU	LU	LU	L
3	U	LU	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

Do nothing

Delete C

Substitute B with D

Substitute D with C

Do nothing

Do nothing

Insert A

	j	0	1	2	3	4	5	6
i	0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU	
2	U	LU	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

Example of Optimal Solution Construction

	1	2	3	4	5	6	7
X_i	A	B	C	B	D	A	B
Y_j	B	D	C	A	B	A	

Operations:

Delete A

Do nothing

Delete C

Substitute B with D

Substitute D with C

Do nothing

Do nothing

Insert A

	j	0	1	2	3	4	5	6
i	0	L	L	L	L	L	L	L
1	U	LU	LU	LU	LU	L	LU	LU
2	U	LU	LU	LU	LU	LU	LU	L
3	U	U	LU	LU	L	L	LU	
4	U	LU	LU	LU	LU	LU	LU	L
5	U	U	LU	LU	LU	LU	LU	LU
6	U	U	U	LU	LU	LU	LU	LU
7	U	LU	U	LU	LU	LU	LU	L

