动态规划篇：最长公共子序列问题

问题背景：子序列

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果

问题背景：子序列

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

问题背景：子序列

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

  - X的子序列

X_1	A	B	C	B	D	A	B

问题背景：子序列

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

  - X的子序列

X_1	A	B	C	B	D	A	B

X_2	A	B	C	B

问题背景：子序列

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

  - X的子序列

X_1	A	B	C	B	D	A	B

X_2	A	B	C	B

X_3	A	C	B	B

问题背景：公共子序列

- 给定两个序列X和Y

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

问题背景：公共子序列

- 给定两个序列X和Y

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

- 公共子序列示例

X_1	C	A

Y_1	C	A

问题背景：公共子序列

- 给定两个序列X和Y

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

- 公共子序列示例

X_1	C	A

Y_1	C	A

X_2	A	B	A

Y_2	A	B	A

问题背景：公共子序列

- 给定两个序列X和Y

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

- 公共子序列示例

X_1	C	A

Y_1	C	A

X_2	A	B	A

Y_2	A	B	A

X_3	B	C	A	B

Y_3	B	C	A	B

问题背景：公共子序列

- 给定两个序列X和Y

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

- 公共子序列示例

X_1	C	A

Y_1	C	A

X_2	A	B	A

Y_2	A	B	A

X_3	B	C	A	B

Y_3	B	C	A	B

问题：如何求两个给定序列的最长公共子序列?

问题定义

- 形式化定义

最长公共子序列问题

Longest Common Subsequence Problem

输入

- 序列$X = < x_1, x_2, \ldots, x_n >$和序列$Y = < y_1, y_2, \ldots, y_m >$

问题定义

- 形式化定义

最长公共子序列问题

Longest Common Subsequence Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子序列$Z = < z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

问题定义

- 形式化定义

最长公共子序列问题

Longest Common Subsequence Problem

输入

- 序列$X =< x_1, x_2, \dots, x_n >$和序列$Y =< y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子序列$Z =< z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

$$s.t. < z_1, z_2, \dots, z_l > =< x_{i_1}, x_{i_2}, \dots, x_{i_l} > =< y_{j_1}, y_{j_2}, \dots, y_{j_l} >$$
$$(1 \le i_1 < i_2, \dots, i_l \le n; 1 \le j_1 < j_2, \dots, j_l \le m)$$

问题定义

- 形式化定义

最长公共子序列问题

Longest Common Subsequence Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子序列$Z = < z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

优化目标

$$s.t. < z_1, z_2, \dots, z_l > = < x_{i_1}, x_{i_2}, \dots, x_{i_l} > = < y_{j_1}, y_{j_2}, \dots, y_{j_l} >$$

$$(1 \le i_1 < i_2, \dots, i_l \le n; 1 \le j_1 < j_2, \dots, j_l \le m)$$

问题定义

- 形式化定义

最长公共子序列问题

Longest Common Subsequence Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子序列$Z = < z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

优化目标

$$s.t. < z_1, z_2, \dots, z_l > = < x_{i_1}, x_{i_2}, \dots, x_{i_l} > = < y_{j_1}, y_{j_2}, \dots, y_{j_l} >$$
$$(1 \le i_1 < i_2, \dots, i_l \le n; 1 \le j_1 < j_2, \dots, j_l \le m)$$

约束条件

蛮力枚举

- 枚举所有子序列

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

X	*A*

枚举并检查长度为1的子序列

蛮力枚举

- **枚举所有子序列**

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

X	A

X	B

枚举并检查长度为1的子序列

蛮力枚举

- **枚举所有子序列**

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

X	*A*

X	*B*

X	*C*

枚举并检查长度为1的子序列

蛮力枚举

- **枚举所有子序列**

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

X	A

X	B

X	C

X	B

X	D

X	A

X	B

枚举并检查长度为1的子序列

蛮力枚举

- 枚举所有子序列

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

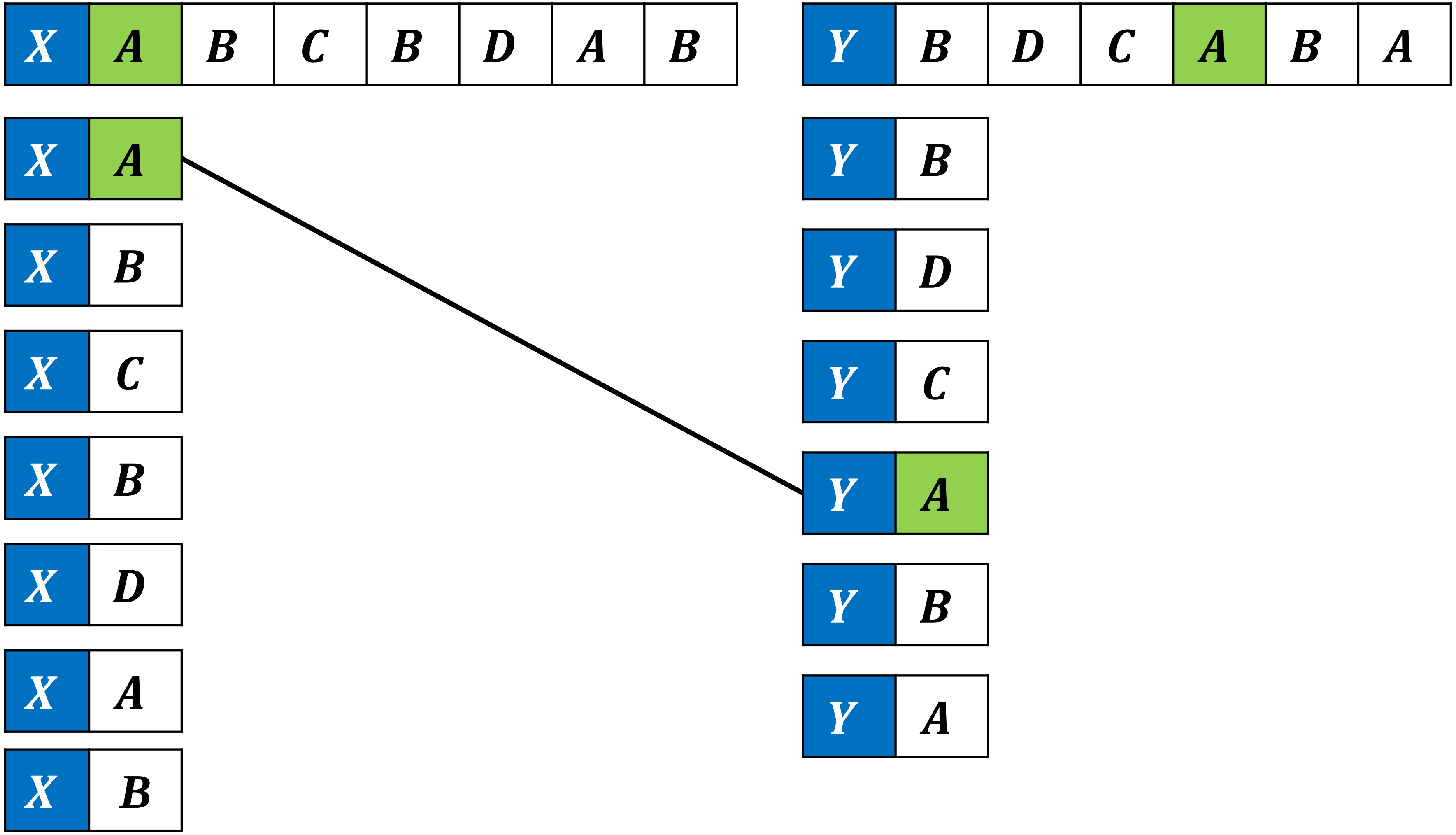
X	*A*
X	*B*
X	*C*
X	*B*
X	*D*
X	*A*
X	*B*

Y	*B*
Y	*D*
Y	*C*
Y	*A*
Y	*B*
Y	*A*

枚举并检查长度为1的子序列

蛮力枚举

- 枚举所有子序列



枚举并检查长度为1的子序列

蛮力枚举

- **枚举所有子序列**

X	A	B	C	B	D	A	B

X	A	B	C	B
X	A	B	C	D
X	A	B	C	A

…

X	B	D	A	B

…

X	C	B	D	B
X	B	D	A	B

Y	B	D	C	A	B	A

Y	B	D	C	A
Y	B	D	C	B

…

Y	B	D	A	B

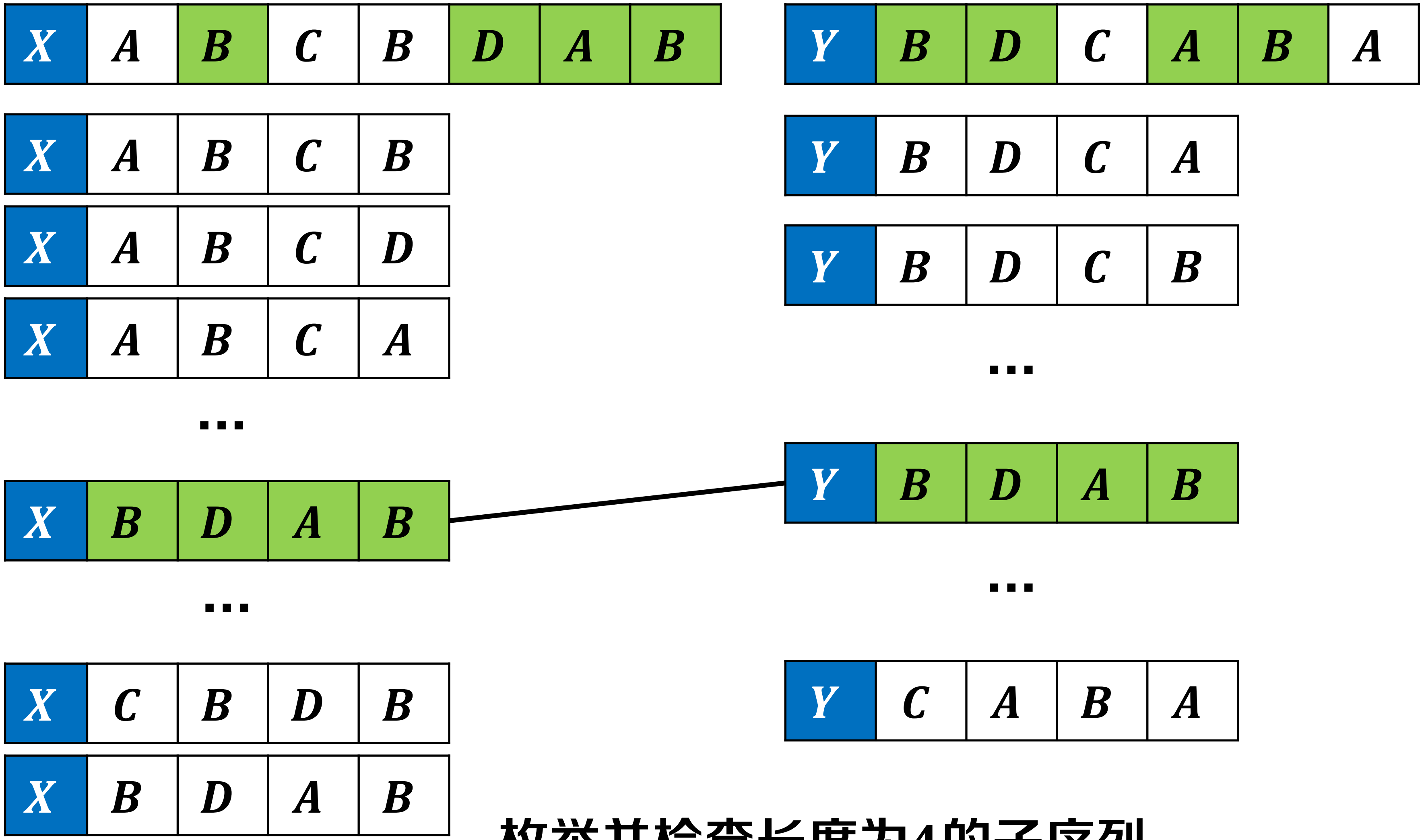
…

Y	C	A	B	A

枚举并检查长度为4的子序列

蛮力枚举

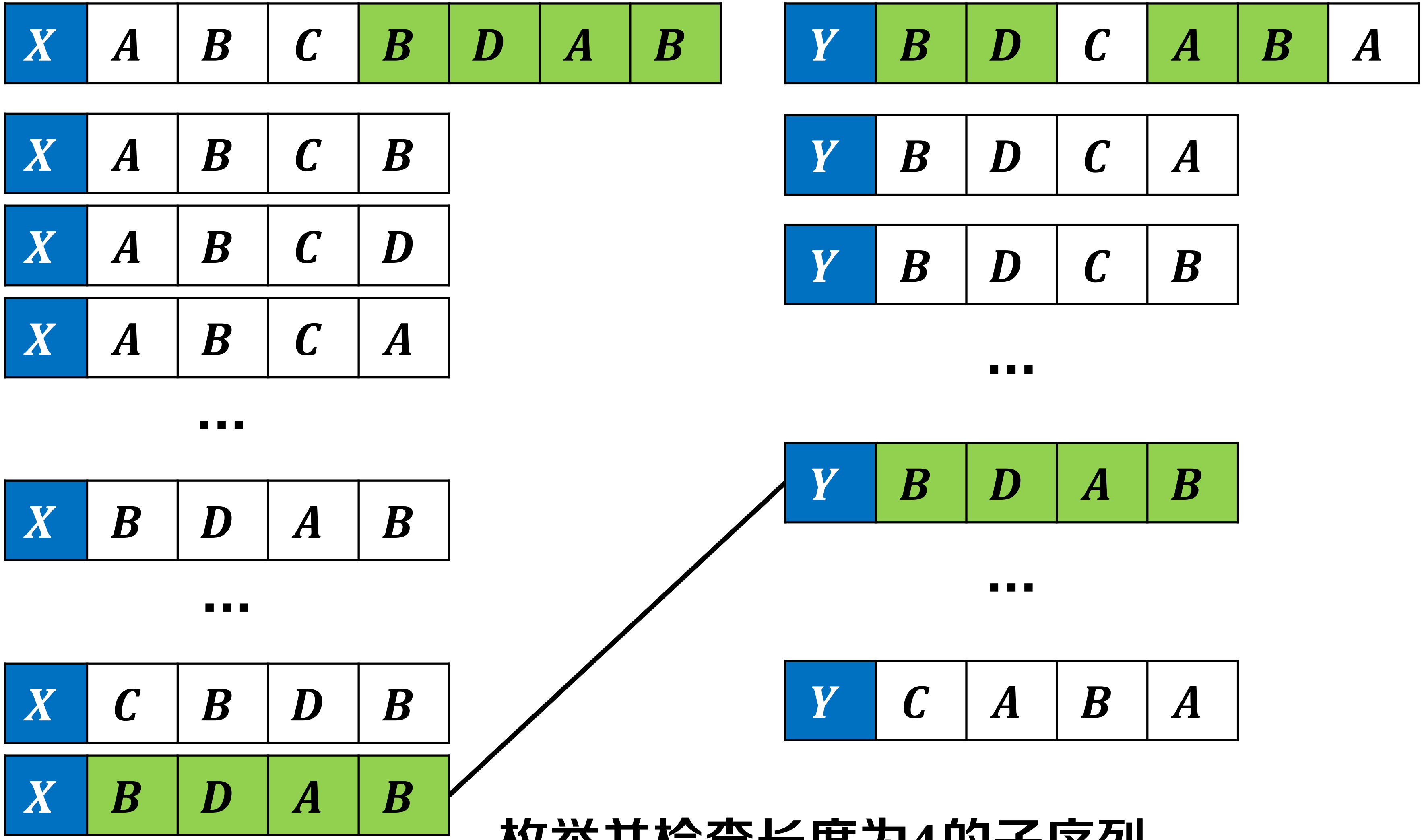
- 枚举所有子序列



枚举并检查长度为4的子序列

蛮力枚举

- 枚举所有子序列



枚举并检查长度为4的子序列

蛮力枚举

- 枚举所有子序列

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

X	*A*	*B*	*C*	*B*	*D*
X	*A*	*B*	*C*	*B*	*A*
X	*A*	*B*	*C*	*B*	*B*
X	*A*	*B*	*B*	*D*	*A*
X	*A*	*B*	*B*	*D*	*B*

…

X	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*
Y	*B*	*D*	*C*	*A*	*A*
Y	*B*	*D*	*C*	*B*	*A*
Y	*B*	*D*	*A*	*B*	*A*
Y	*B*	*C*	*A*	*B*	*A*
Y	*D*	*C*	*A*	*B*	*A*

枚举并检查长度为5的子序列

蛮力枚举

- 枚举所有子序列

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

X	*A*	*B*	*C*	*B*	*D*
X	*A*	*B*	*C*	*B*	*A*
X	*A*	*B*	*C*	*B*	*B*
X	*A*	*B*	*B*	*D*	*A*
X	*A*	*B*	*B*	*D*	*B*

…

X	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*
Y	*B*	*D*	*C*	*A*	*A*
Y	*B*	*D*	*C*	*B*	*A*
Y	*B*	*D*	*A*	*B*	*A*
Y	*B*	*C*	*A*	*B*	*A*
Y	*D*	*C*	*A*	*B*	*A*

枚举并检查长度为5的子序列

蛮力枚举

- 枚举所有子序列

X	Y	
X A B C B D A B	Y B D C A B A	
X B	Y B	长度为1
X A B	Y A B	长度为2
X A B A	Y A B A	长度为3
X B D A B	Y B D A B	长度为4
X □ □ □ □ □	Y □ □ □ □ □	长度为5
X □ □ □ □ □ □	Y □ □ □ □ □ □	长度为6

蛮力枚举

- 枚举所有子序列

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*

Y	*B*	*D*	*C*	*A*	*B*	*A*

长度为1

X	*B*
Y	*B*

长度为2

X	*A*	*B*
Y	*A*	*B*

长度为3

X	*A*	*B*	*A*
Y	*A*	*B*	*A*

长度为4

X	*B*	*D*	*A*	*B*
Y	*B*	*D*	*A*	*B*

蛮力枚举

- **枚举所有子序列**

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

X	B	Y	B

长度为1

X	A	B	Y	A	B

长度为2

X	A	B	A	Y	A	B	A

长度为3

X	B	D	A	B	Y	B	D	A	B

长度为4

最长公共子序列

枚举观察

X	B	D	A	B

Y	B	D	A	B

长度为4

枚举观察

X B D A B	Y B D A B	长度为4
X D A B	Y D A B	长度为3

枚举观察

X	*B*	*D*	*A*	*B*	*Y*	*B*	*D*	*A*	*B*	长度为4
X	*D*	*A*	*B*		*Y*	*D*	*A*	*B*		长度为3
X	*A*	*B*			*Y*	*A*	*B*			长度为2

枚举观察

X *B* *D* *A* *B*	*Y* *B* *D* *A* *B*	长度为4
X *D* *A* *B*	*Y* *D* *A* *B*	长度为3
X *A* *B*	*Y* *A* *B*	长度为2
X *B*	*Y* *B*	长度为1

枚举观察

X B D A B	Y B D A B	长度为4
X D A B	Y D A B	长度为3
X A B	Y A B	长度为2
X B	Y B	长度为1

- 可能存在**最优子结构**和**重叠子问题**

枚举观察

X B D A B	Y B D A B	长度为4
X D A B	Y D A B	长度为3
X A B	Y A B	长度为2
X B	Y B	长度为1

- 可能存在**最优子结构**和**重叠子问题**

问题：如何利用动态规划求解？

问题结构分析

- 给出问题表示
  - $C[i,j]$：$X[1..i]$和$Y[1..j]$的最长公共子序列长度

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

问题结构分析

- **给出问题表示**
  - $C[i,j]$：$X[1..i]$和$Y[1..j]$的最长公共子序列长度

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- **明确原始问题**
  - $C[n,m]$： $X[1..n]$和$Y[1..m]$的最长公共子序列长度

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况1：$x_7 \neq y_6$

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*
	Y	*B*	*D*	*C*	*A*	*B*	*A*

  - 情况2：$x_7 = y_6$

X	*A*	*B*	*C*	*B*	*D*	*A*	*B*
		Y	*B*	*D*	*C*	*A*	*B*

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	B	D	A	B
	Y	B	D	C	A	B	A

不同时出现

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符

  - 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	A

$C[7,6-1]+0$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

$C[7,6-1]+0$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

$C[7-1,6]+0$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

max

$C[7,6-1]+0$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

$C[7-1,6]+0$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	A	

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i \neq y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i,j-1]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i \neq y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i,j-1]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- $C[i,j] = \max\{C[i-1,j], C[i,j-1]\}$

递推关系建立：分析最优（子）结构

- $x_i \neq y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i,j-1]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- $\boxed{C[i,j]} = \max\{\boxed{C[i-1,j], C[i,j-1]}\}$

最优子结构

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

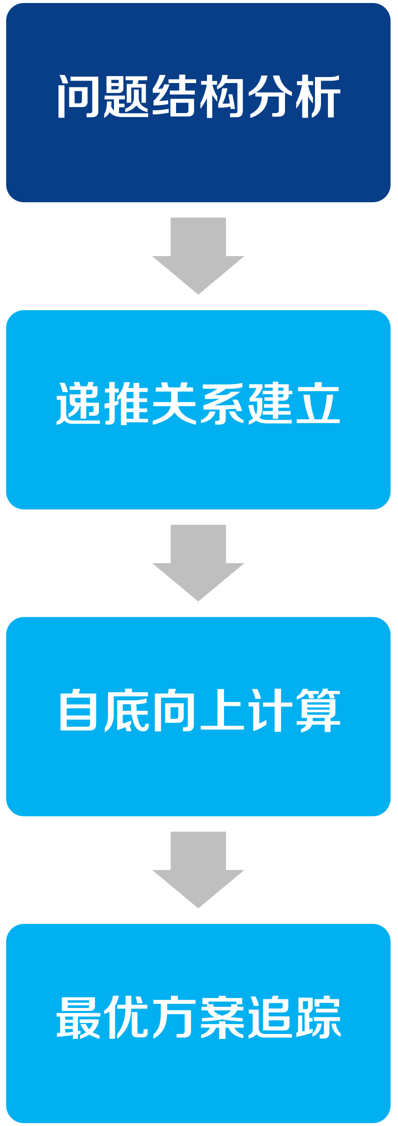
- 考察末尾字符

  - 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	B	D	A	B
	Y	B	D	C	A	B	B

可同时出现



递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况2：$x_7 = y_6$

$C[7-1, 6-1] + 1$

X	A	B	C	B	D	A	B
Y	B	D	C	A	B	B	

$C[7, 6]$

X	A	B	C	B	D	A	B
	Y	B	D	C	A	B	B

可同时出现

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

$C[7-1,6-1]+1$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

也可不同时出现

$C[7,6-1]+0$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

$C[7-1,6]+0$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾字符
  - 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

max

$C[7-1,6-1]+1$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

$C[7,6-1]+0$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

$C[7-1,6]+0$

X	A	B	C	B	D	A	B

Y	B	D	C	A	B	B

问题结构分析

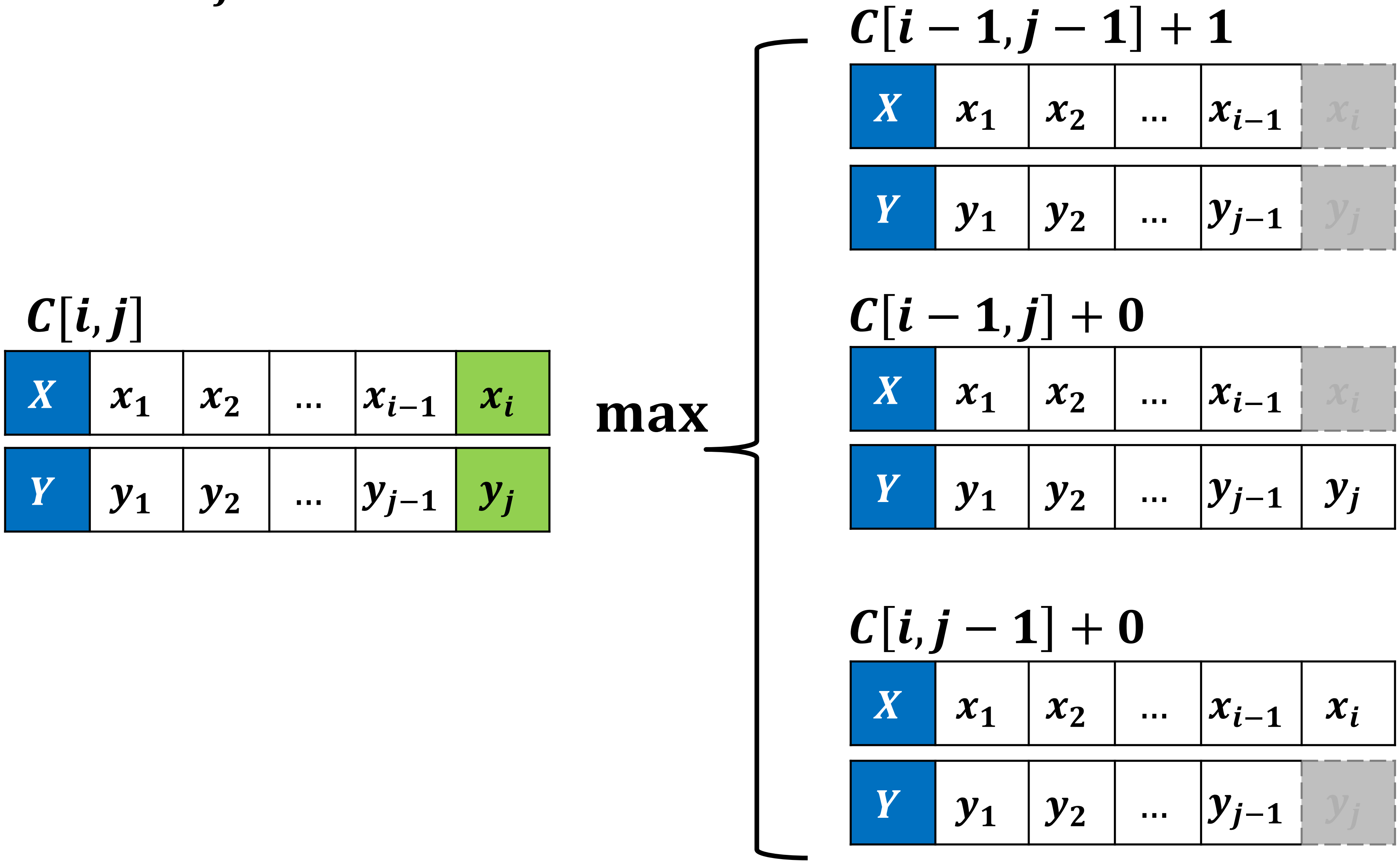
递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$



递推关系建立：分析最优（子）结构

- $x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i-1,j-1]+1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i,j-1]+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题：3个问题是否都需要求解?

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$
  - $C[i-1,j]$比$C[i-1,j-1]$至多大1
  - $C[i,j-1]$比$C[i-1,j-1]$至多大1

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$$\max\begin{cases} \boxed{C[i-1,j-1]}+1 \\ \boxed{C[i-1,j]}+0 \\ \boxed{C[i,j-1]}+0 \end{cases}$$

$\boxed{C[i-1,j-1]}+1$

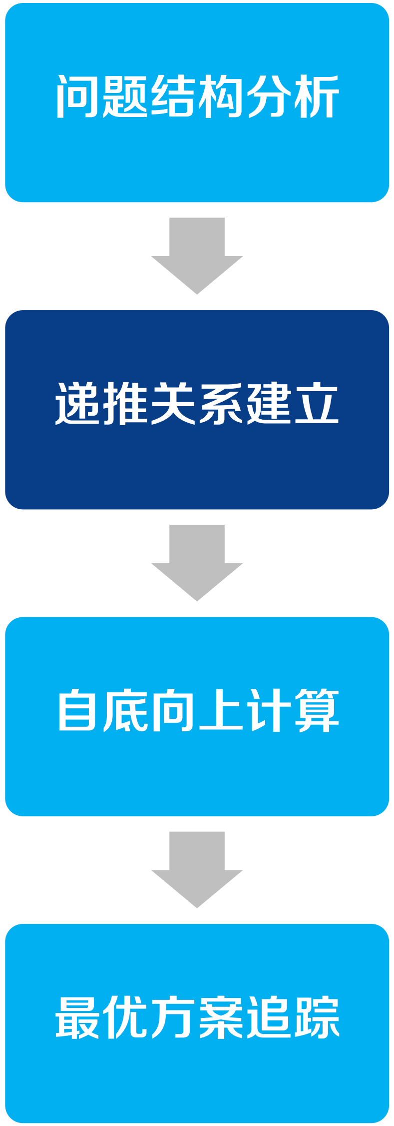
X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$\boxed{C[i-1,j]}+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$\boxed{C[i,j-1]}+0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j



递推关系建立：分析最优（子）结构

- $x_i = y_j$
  - $C[i-1,j]$比$C[i-1,j-1]$至多大1
  - $C[i,j-1]$比$C[i-1,j-1]$至多大1
  - $C[i-1,j-1]+1$，另外两个+0

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i-1,j-1] + 1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j] + 0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i,j-1] + 0$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$
  - $C[i-1,j]$比$C[i-1,j-1]$至多大1
  - $C[i,j-1]$比$C[i-1,j-1]$至多大1
  - $C[i-1,j-1]+1$，另外两个+0

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i-1,j-1]\boxed{+1}$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]\boxed{+0}$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i,j-1]\boxed{+0}$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$$C[i-1,j-1]+1 \geq \max\{C[i,j-1],C[i-1,j]\}$$

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$
  - $C[i-1,j]$比$C[i-1,j-1]$至多大1
  - $C[i,j-1]$比$C[i-1,j-1]$至多大1
  - $C[i-1,j-1]+1$，另外两个+0

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

max

$C[i-1,j-1]+1$ 已充分

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j]+0$ 非必要

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i,j-1]+0$ 非必要

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$$C[i-1,j-1]+1 \geq \max\{C[i,j-1], C[i-1,j]\}$$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$\longrightarrow$

$C[i-1,j-1]+1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$\longrightarrow$

$C[i-1,j-1]+1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- $C[i,j] = C[i-1,j-1]+1$

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- $x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i-1,j-1]+1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- $\boxed{C[i,j]} = \boxed{C[i-1,j-1]} + 1$

最优子结构

问题结构分析

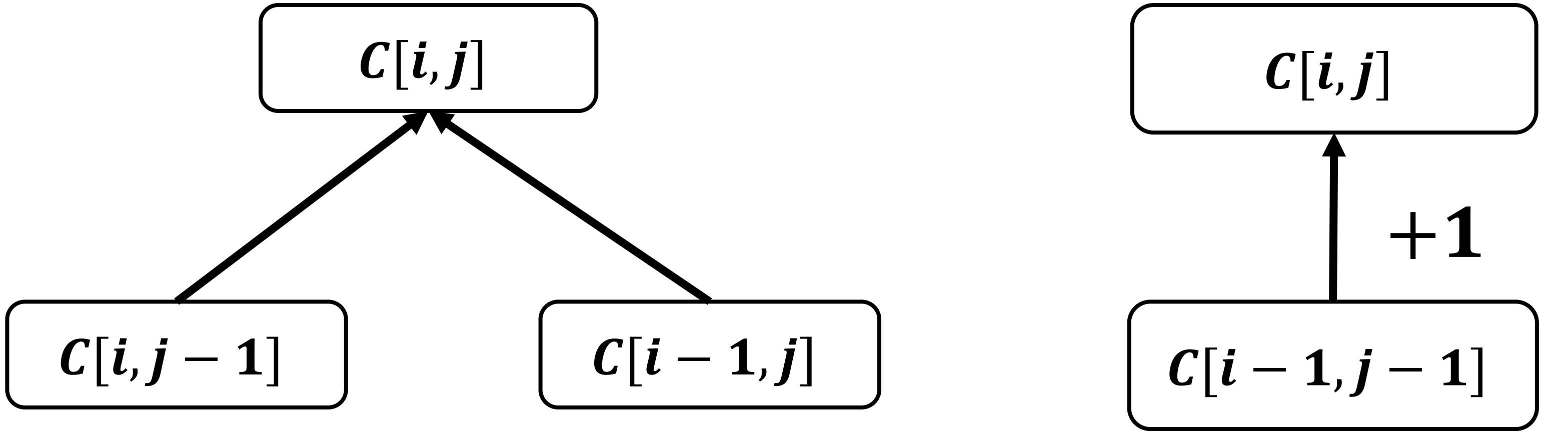
递推关系建立

自底向上计算

最优方案追踪

递推关系建立：构造递推公式

- $$C[i,j]=\begin{cases}\max\{C[i-1,j],C[i,j-1]\}, & x_i\neq y_j\\ C[i-1,j-1]+1 & ,x_i=y_j\end{cases}$$



自底向上计算：确定计算顺序

- **初始化**
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…					
$i=n$					

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- **初始化**
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0				
$i=n$	0				

初始化

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- **初始化**
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0
- **递推公式**
  - $C[i,j] = \begin{cases} \max\{C[i-1,j], C[i,j-1]\}, x_i \neq y_j \\ C[i-1,j-1]+1 \qquad ,x_i = y_j \end{cases}$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0			$C[i,j]$	
$i=n$	0				

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0
- 递推公式
  - $C[i,j] = \begin{cases} \max\{C[i-1,j], C[i,j-1]\}, & x_i \neq y_j \\ C[i-1,j-1]+1 & , x_i = y_j \end{cases}$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0			$C[i,j]$	
$i=n$	0				

问题结构分析 → 递推关系建立 → 自底向上计算 → 最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0
- 递推公式
  - $$C[i,j] = \begin{cases} \max\{C[i-1,j], C[i,j-1]\}, & x_i \neq y_j \\ C[i-1,j-1]+1 & , x_i = y_j \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0			$C[i,j]$	
$i=n$	0				

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0]=C[0,j]=0$
    - 某序列长度为0时，最长公共子序列长度为0
- 递推公式
  - $$C[i,j]=\begin{cases}\max\{C[i-1,j],C[i,j-1]\}, & x_i\neq y_j\\ C[i-1,j-1]+1 & ,x_i=y_j\end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0			$C[i,j]$	
$i=n$	0				

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：依次求解问题

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子序列长度为0
- 递推公式
  - $C[i,j] = \begin{cases} \max\{C[i-1,j], C[i,j-1]\}, x_i \neq y_j \\ C[i-1,j-1]+1 \qquad ,x_i = y_j \end{cases}$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0				
$i=n$	0				★

自底向上计算

问题结构分析 → 递推关系建立 → 自底向上计算 → 最优方案追踪

最优方案追踪：记录决策过程

- 构造追踪数组$rec[1..n]$，记录子问题来源

  - $$rec[i,j]=\begin{cases}LU, & if\ C[i,j]=C[i-1,j-1]+1\\ U, & if\ C[i,j]=C[i-1,j]\\ L, & if\ C[i,j]=C[i,j-1]\end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	$\dots$	$j=m$
$i=0$					
$i=1$					
$i=2$					
$\dots$				$C[i,j]$	
$i=n$					

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

- 构造追踪数组$rec[1..n]$，记录子问题来源

  - $$rec[i,j]=\begin{cases}LU, & if\ C[i,j]=C[i-1,j-1]+1\\ U, & if\ C[i,j]=C[i-1,j]\\ L, & if\ C[i,j]=C[i,j-1]\end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…				$C[i,j]$	
$i=n$					

  - 最长公共子序列末尾为$X[i]=Y[j]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

- 构造追踪数组$rec[1..n]$，记录子问题来源
  - $$rec[i,j]=\begin{cases}LU, & if\ C[i,j]=C[i-1,j-1]+1\\ U, & if\ C[i,j]=C[i-1,j]\\ L, & if\ C[i,j]=C[i,j-1]\end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…				$C[i,j]$	
$i=n$					

  - 最长公共子序列在$X[1..i-1]$和$Y[1..j]$中

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

- 构造追踪数组$rec[1..n]$，记录子问题来源

  - $$rec[i,j]=\begin{cases}LU, & if\ C[i,j]=C[i-1,j-1]+1\\ U, & if\ C[i,j]=C[i-1,j]\\ L, & if\ C[i,j]=C[i,j-1]\end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…				$C[i,j]$	
$i=n$					

  - 最长公共子序列在$X[1..i]$和$Y[1..j-1]$中

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 输出最长公共子序列

  - $$rec[i,j]=\begin{cases} LU, & if\ C[i,j]=C[i-1,j-1]+1 \\ U, & if\ C[i,j]=C[i-1,j] \\ L, & if\ C[i,j]=C[i,j-1] \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…					
$i=n$					

$rec[\ ]=L$

  - 最长公共子序列在$X[1..i]$和$Y[1..j-1]$中

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- **输出最长公共子序列**

  - $$rec[i,j]=\begin{cases} LU, & if\ C[i,j]=C[i-1,j-1]+1 \\ U, & if\ C[i,j]=C[i-1,j] \\ L, & if\ C[i,j]=C[i,j-1] \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…					
$i=n$					

$rec[\,]=U$

$rec[\,]=L$

  - 最长公共子序列在$X[1..i-1]$和$Y[1..j]$中

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 输出最长公共子序列

  - $$rec[i,j]=\begin{cases} LU, & if\ C[i,j]=C[i-1,j-1]+1 \\ U, & if\ C[i,j]=C[i-1,j] \\ L, & if\ C[i,j]=C[i,j-1] \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$				$rec[\ ]=LU$	
…					
$i=n$			$rec[\ ]=U$		

$rec[\ ]=L$

  - 最长公共子序列末尾为$X[i]=Y[j]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：输出最优方案

- 输出最长公共子序列

  - $$rec[i,j]=\begin{cases} LU, & if\ C[i,j]=C[i-1,j-1]+1 \\ U, & if\ C[i,j]=C[i-1,j] \\ L, & if\ C[i,j]=C[i,j-1] \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$					
$i=1$					
$i=2$					
…					
$i=n$					

$rec[\,]=LU$

$rec[\,]=U$

$rec[\,]=L$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

算法实例

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

$C[\,]$

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

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

初始化

$rec[\,]$

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

算法实例

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

$X_i \neq Y_j$

$C[\,]$

j / i	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						

$rec[\,]$

j / i	1	2	3	4	5	6
1						
2						
3						
4						
5						
6						
7						

算法实例

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

$X_i \neq Y_j$

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$X_i \neq Y_j$

$C[\,]$

$i \backslash j$	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						

$\mathrm{C}[1,1] = \max\{C[1,0], C[0,1]\}$

$rec[\,]$

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

算法实例

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

$C[\,]$

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

$rec[\,]$

i \ j	1	2	3	4	5	6
1	U	U				
2						
3						
4						
5						
6						
7						

算法实例

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

$C[\,]$

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

$rec[\,]$

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

算法实例

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

$X_i = Y_j$

$C[\,]$

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

$C[1,4] = C[0,3] + 1$

$rec[\,]$

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

算法实例

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

$C[\,]$

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

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

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

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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				
4	0						
5	0						
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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		
4	0						
5	0						
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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	
4	0						
5	0						
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						
5	0						
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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				
5	0						
6	0						
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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						

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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		
5	0						
6	0						
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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		
5						
6						
7						

算法实例

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

$C[\,]$

$i \backslash j$	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	
5	0						
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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					
6	0						
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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						

算法实例

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

$C[\,]$

i \ 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				
6	0						
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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						

算法实例

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

$C[\,]$

$i \backslash j$	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			
6	0						
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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						
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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						

算法实例

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

$C[\,]$

$i \backslash j$	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					
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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				
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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						

算法实例

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

$C[\,]$

i \ 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			
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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			
7						

算法实例

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

$C[\,]$

$i \backslash j$	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		
7	0						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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	
7	0						

$rec[\,]$

i \ j	1	2	3	4	5	6
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	
7						

算法实例

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

$C[\,]$

$i \backslash j$	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						

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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					

$rec[\,]$

i \ j	1	2	3	4	5	6
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					

算法实例

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

$C[\,]$

$i \backslash j$	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				

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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			

$rec[\,]$

i \ j	1	2	3	4	5	6
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			

算法实例

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

$C[\,]$

$i \backslash j$	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		

$rec[\,]$

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

算法实例

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

$C[\,]$

$i \backslash j$	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	

$rec[\,]$

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

算法实例

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

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

算法实例

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

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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
[illegible]	[illegible]	[illegible]	U	U	LU	U

最长公共子序列的长度

算法实例

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

$C[\,]$

$i \backslash j$	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

$rec[\,]$

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

算法实例

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

			A

$C[\,]$

$i \backslash j$	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

$rec[\,]$

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

算法实例

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

			A

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

算法实例

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

		B	A

$C[\,]$

$i \backslash j$	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

$rec[\,]$

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

算法实例

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

		B	A

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

算法实例

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

	C	B	A

$C[\,]$

$i \backslash j$	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

$rec[\,]$

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

算法实例

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

	C	B	A

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

算法实例

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

B	C	B	A

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

算法实例

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

B	C	B	A

最长公共子序列

$C[\,]$

i \ 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

$rec[\,]$

i \ j	1	2	3	4	5	6
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

伪代码

- **Longest-Common-Subsequence(X, Y)**

输入: 两个序列X,Y
输出: X和Y的最长公共子序列

$n \leftarrow \text{length}(X)$
$m \leftarrow \text{length}(Y)$

序列长度

//初始化

新建二维数组$C[0..n, 0..m]$和$rec[0..n, 0..m]$

for $i \leftarrow 0$ *to* n **do**
| $C[i, 0] \leftarrow 0$
end
for $j \leftarrow 0$ *to* m **do**
| $C[0, j] \leftarrow 0$
end

伪代码

- **Longest-Common-Subsequence(X, Y)**

输入: 两个序列X,Y
输出: X和Y的最长公共子序列
$n \leftarrow$length(X)
$m \leftarrow$length(Y)
//初始化
新建二维数组$C[0..n, 0..m]$和$rec[0..n, 0..m]$
for $i \leftarrow 0$ *to* n **do**
  $C[i, 0] \leftarrow 0$
end
for $j \leftarrow 0$ *to* m **do**
  $C[0, j] \leftarrow 0$
end

初始化

伪代码

- **Longest-Common-Subsequence(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i = Y_j then
            C[i, j] ← C[i − 1, j − 1] + 1
            rec[i, j] ← "LU"
        end
        else if C[i − 1, j] ≥ C[i, j − 1] then
            C[i, j] ← C[i − 1, j]
            rec[i, j] ← "U"
        end
        else
            C[i, j] ← C[i, j − 1]
            rec[i, j] ← "L"
        end
    end
end
return C, rec
```

依次计算子问题

伪代码

- **Longest-Common-Subsequence(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i = Y_j then
            C[i, j] ← C[i − 1, j − 1] + 1
            rec[i, j] ← "LU"
        end
        else if C[i − 1, j] ≥ C[i, j − 1] then
            C[i, j] ← C[i − 1, j]
            rec[i, j] ← "U"
        end
        else
            C[i, j] ← C[i, j − 1]
            rec[i, j] ← "L"
        end
    end
end
return C, rec
```

末尾相等

伪代码

- **Longest-Common-Subsequence(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i = Y_j then
            C[i, j] ← C[i − 1, j − 1] + 1
            rec[i, j] ← "LU"
        end
        else if C[i − 1, j] ≥ C[i, j − 1] then
            C[i, j] ← C[i − 1, j]
            rec[i, j] ← "U"
        end
        else
            C[i, j] ← C[i, j − 1]
            rec[i, j] ← "L"
        end
    end
end
return C, rec
```

记录长度和决策

伪代码

- **Longest-Common-Subsequence(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i = Y_j then
            C[i, j] ← C[i − 1, j − 1] + 1
            rec[i, j] ← "LU"
        end
        else if C[i − 1, j] ≥ C[i, j − 1] then
            C[i, j] ← C[i − 1, j]
            rec[i, j] ← "U"
        end
        else
            C[i, j] ← C[i, j − 1]
            rec[i, j] ← "L"
        end
    end
end
return C, rec
```

末尾不等

伪代码

- **Print-LCS(rec, X, i, j)**

```
输入: 追踪数组rec, 序列X, 当前位置i和j
输出: X[1..i]和Y[1..j]的最长公共子序列
if i = 0 or j = 0 then
 |  return NULL
end
if rec[i, j] = "LU" then
 |  Print-LCS(rec, X, i − 1, j − 1)
 |  print x_i
end
else if rec[i, j] = "U" then
 |  Print-LCS(rec, X, i − 1, j)
end
else
 |  Print-LCS(rec, X, i, j − 1)
end
```

倒序追踪方案

伪代码

- **Print-LCS(rec, X, i, j)**

输入: 追踪数组rec, 序列X, 当前位置i和j
输出: $X[1..i]$和$Y[1..j]$的最长公共子序列

```
if i = 0 or j = 0 then
|   return NULL
end
if rec[i, j] = "LU" then
|   Print-LCS(rec, X, i − 1, j − 1)
|   print x_i
end
else if rec[i, j] = "U" then
|   Print-LCS(rec, X, i − 1, j)
end
else
|   Print-LCS(rec, X, i, j − 1)
end
```

递归终止：序列长度为0

伪代码

- **Print-LCS(rec, X, i, j)**

```
输入: 追踪数组rec, 序列X, 当前位置i和j
输出: X[1..i]和Y[1..j]的最长公共子序列
if i = 0 or j = 0 then
 |  return NULL
end
if rec[i, j] = "LU" then
 |  Print-LCS(rec, X, i − 1, j − 1)
 |  print x_i
end
else if rec[i, j] = "U" then
 |  Print-LCS(rec, X, i − 1, j)
end
else
 |  Print-LCS(rec, X, i, j − 1)
end
```

追踪方案：末尾相等

伪代码

- **Print-LCS(rec, X, i, j)**

输入: 追踪数组rec, 序列X, 当前位置i和j
输出: $X[1..i]$和$Y[1..j]$的最长公共子序列

```
if i = 0 or j = 0 then
|   return NULL
end
if rec[i, j] = "LU" then
|   Print-LCS(rec, X, i − 1, j − 1)
|   print x_i
end
else if rec[i, j] = "U" then
|   Print-LCS(rec, X, i − 1, j)
end
else
|   Print-LCS(rec, X, i, j − 1)
end
```

追踪方案：末尾不等

时间复杂度分析

- **Longest-Common-Subsequence(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i = Y_j then
            C[i, j] ← C[i − 1, j − 1] + 1
            rec[i, j] ← "LU"
        end
        else if C[i − 1, j] ≥ C[i, j − 1] then
            C[i, j] ← C[i − 1, j]
            rec[i, j] ← "U"
        end
        else
            C[i, j] ← C[i, j − 1]
            rec[i, j] ← "L"
        end
    end
end
return C, rec
```

时间复杂度：$O(n \cdot m)$

动态规划篇：编辑距离问题

问题背景

- 输入法自动更正

kittchen

kithen

kitchem

实际输入

kitchen

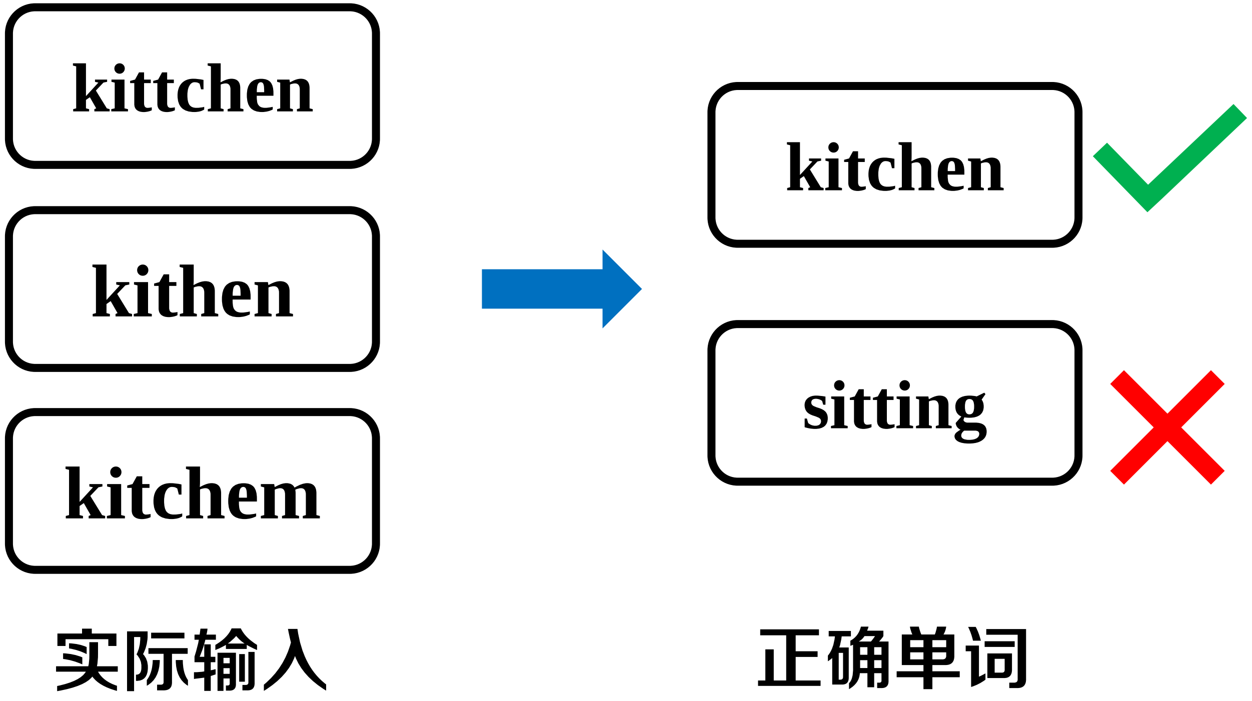
?

sitting

正确单词

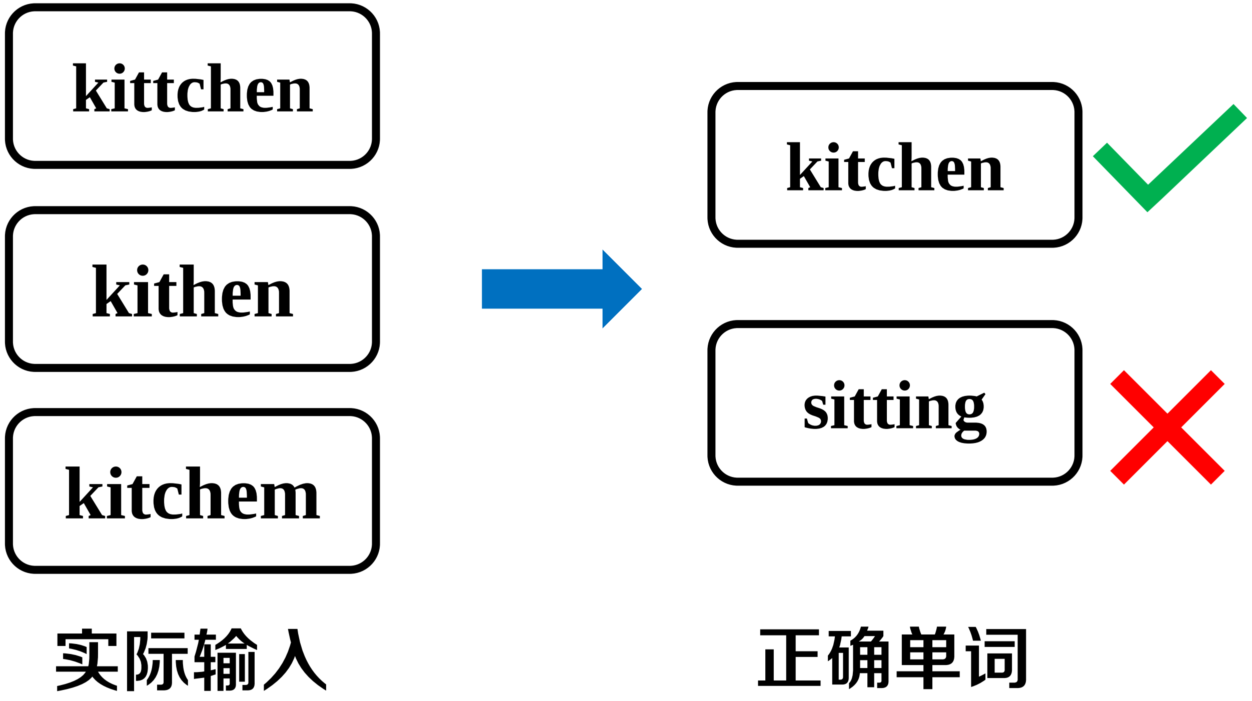
问题背景

- 输入法自动更正



问题背景

- 输入法自动更正



问题：如何衡量序列的相似程度？

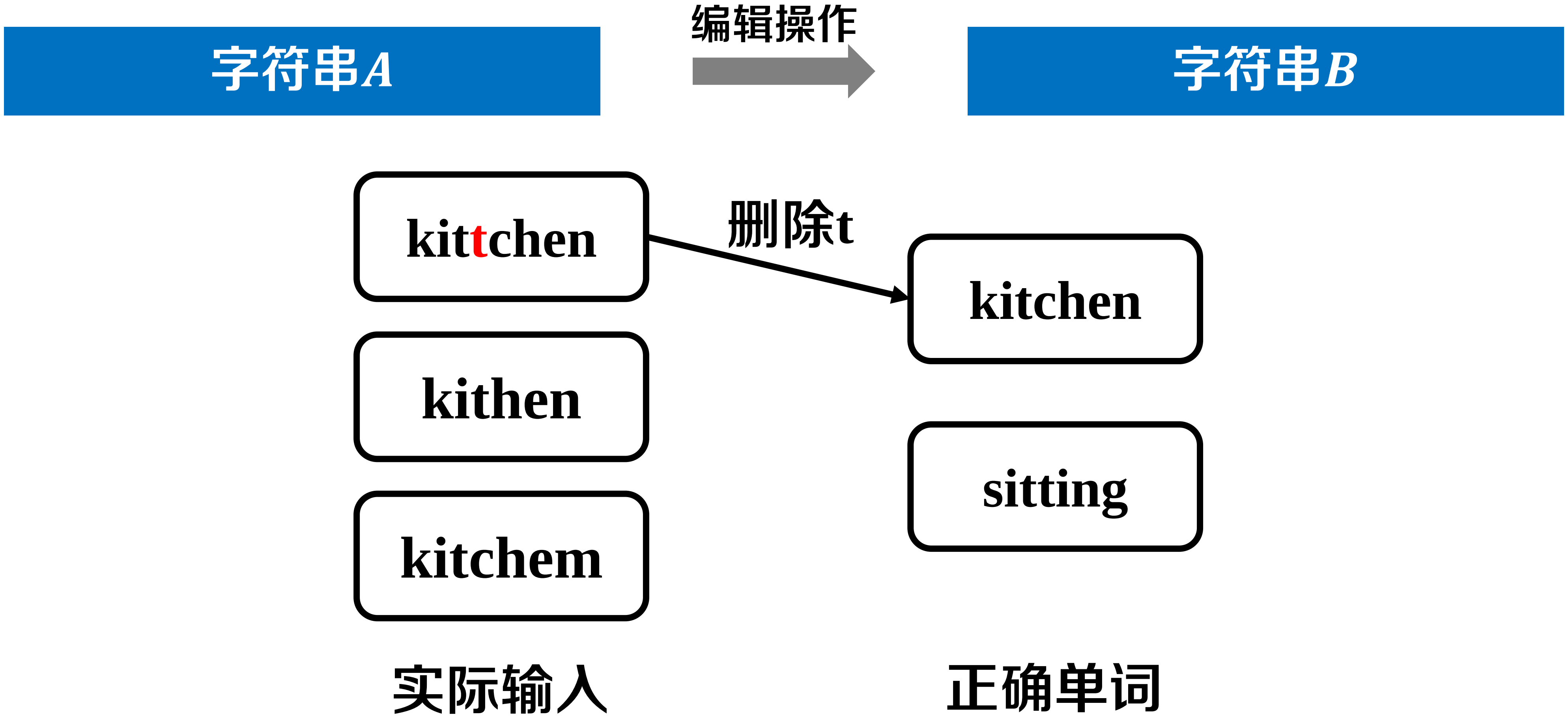
编辑操作

- 基本思想

- 编辑操作：删除、插入、替换

编辑操作

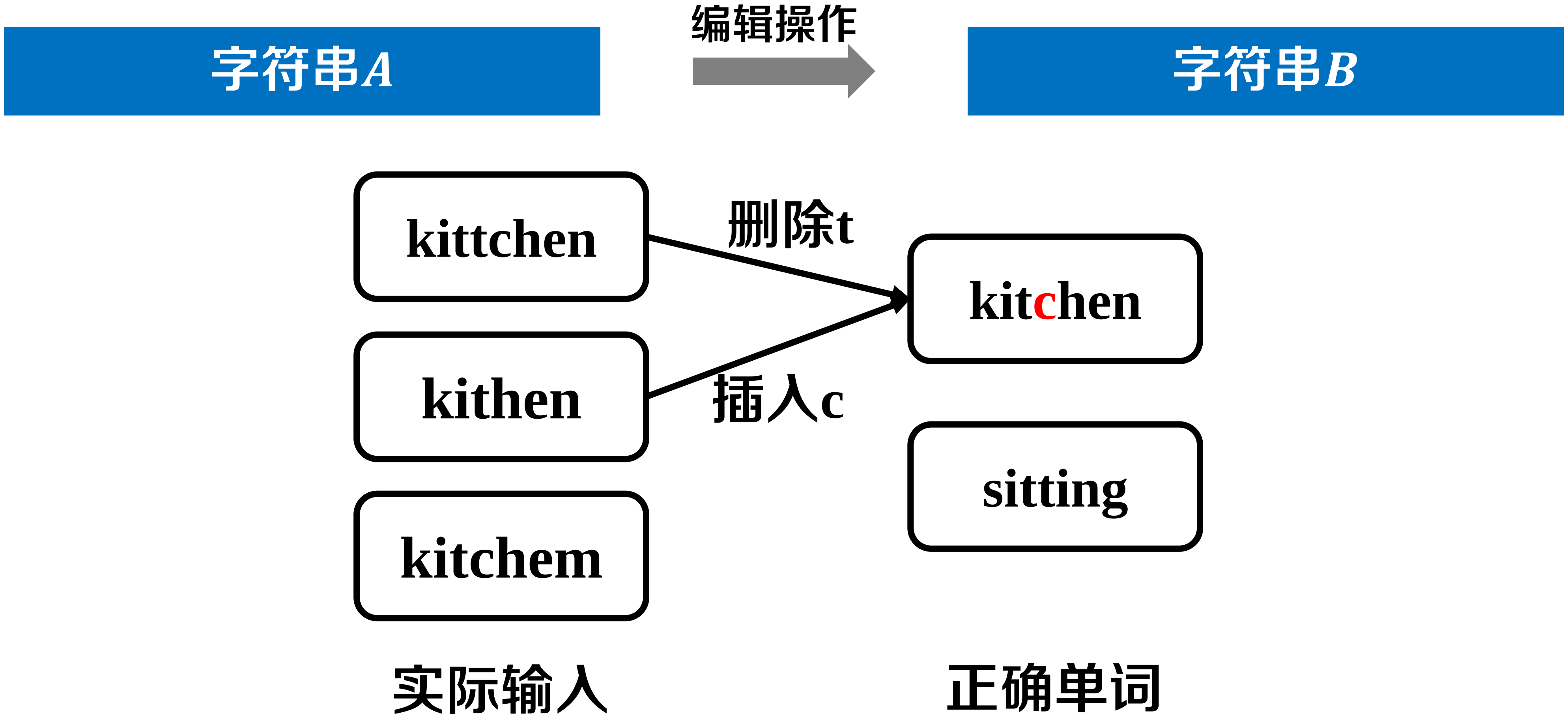
- 基本思想



- 编辑操作：删除、插入、替换

编辑操作

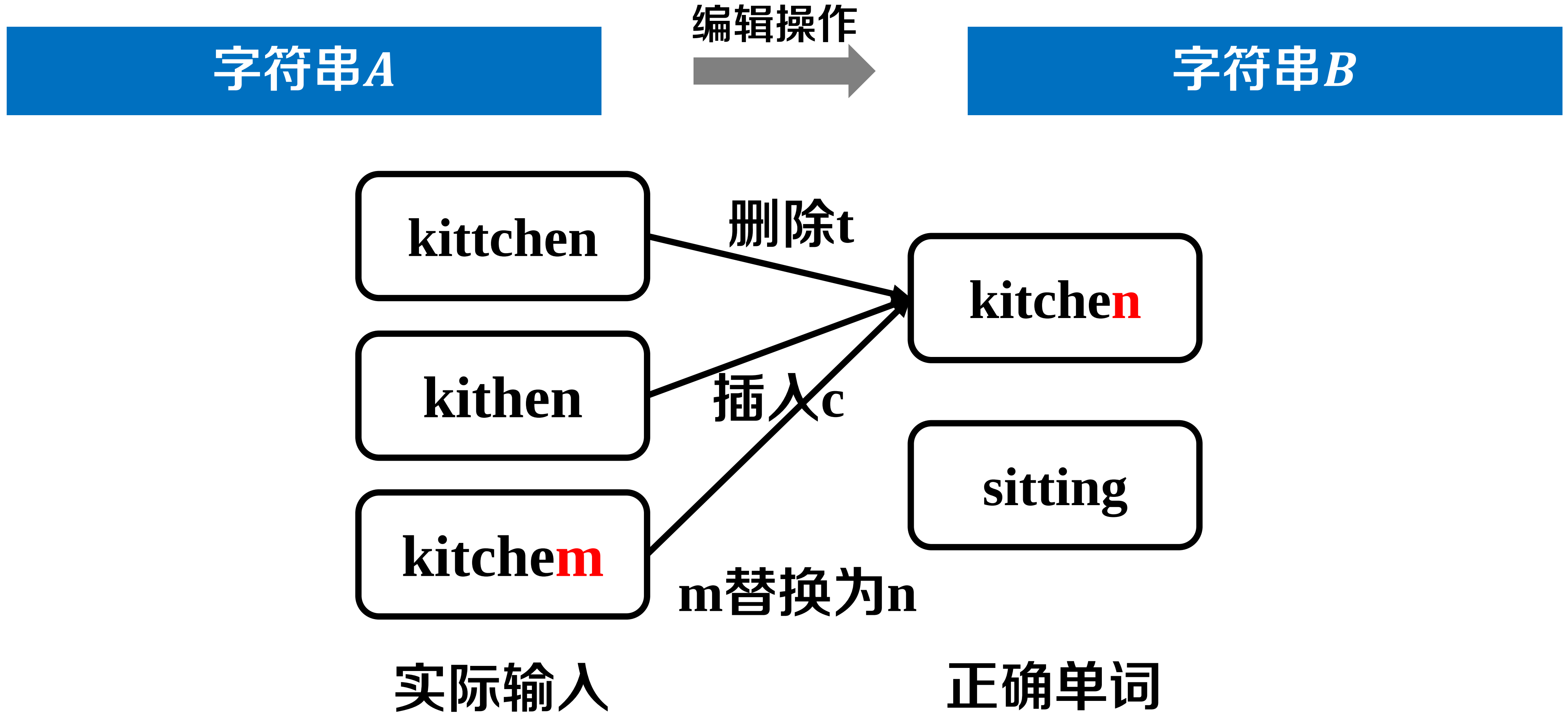
- 基本思想



- 编辑操作：删除、插入、替换

编辑操作

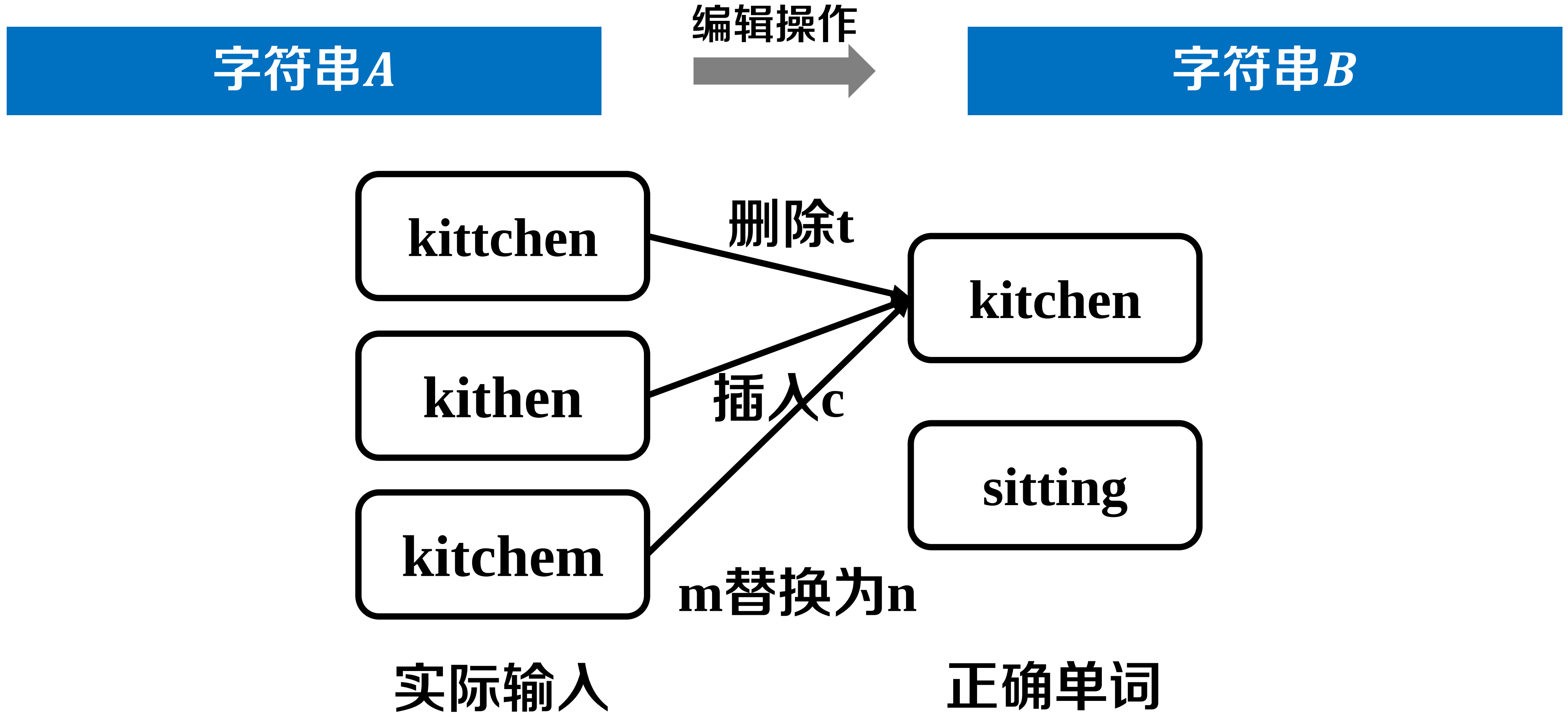
- 基本思想



- 编辑操作：删除、插入、替换

编辑操作

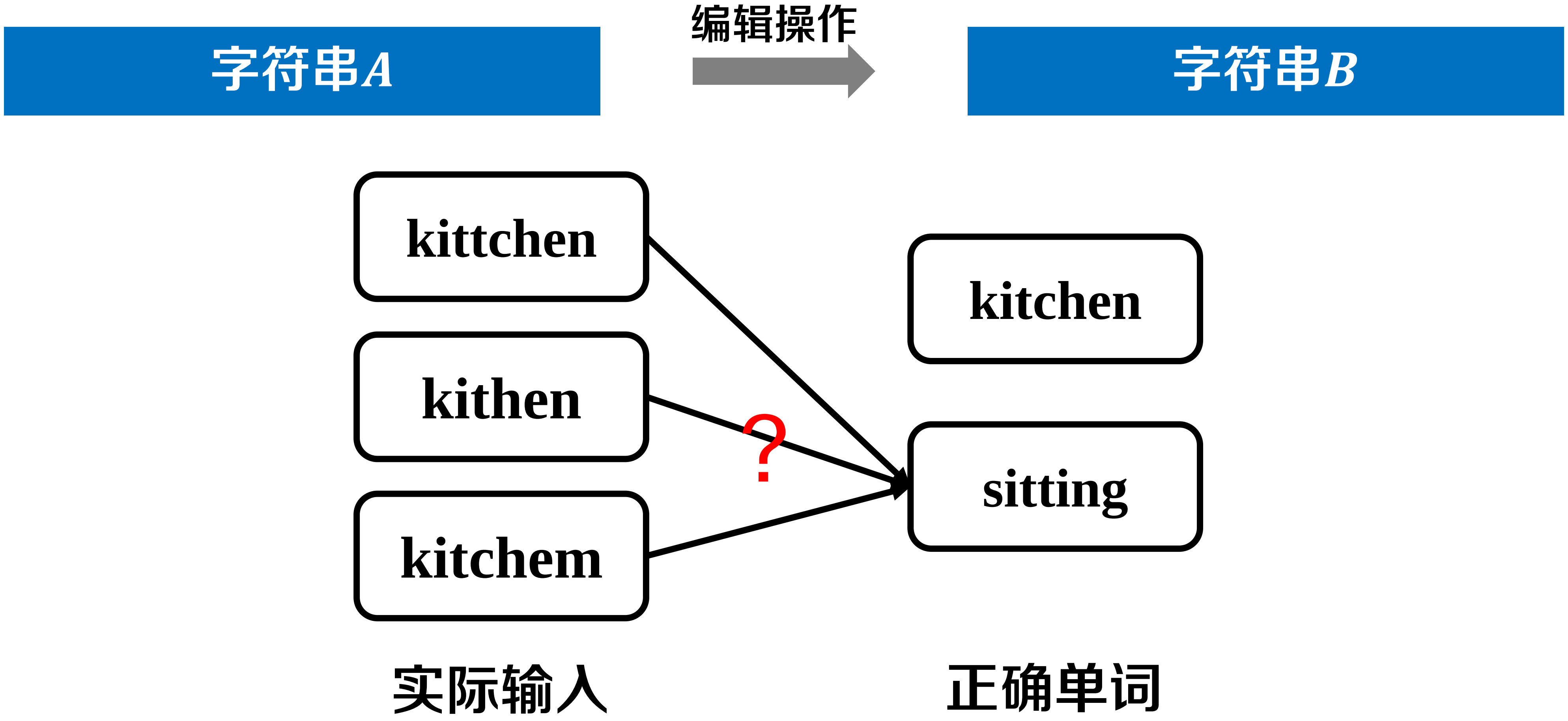
- 基本思想



- 编辑操作：删除、插入、替换

编辑操作

- 基本思想



- 编辑操作：删除、插入、替换

编辑操作示例

A = kittchen

编辑操作

B = sitting

操作名称	操作示例
删除	kittchen → kitchen
插入	kithen → kitchen
替换	kitchem → kitchen

编辑操作示例

A = kittchen

↓ 编辑操作

B = sitting

操作名称	操作示例
删除	kittchen → kitchen
插入	kithen → kitchen
替换	kitchem → kitchen

6次

- 方案1 kittchen $\xrightarrow{k\to s}$ sittchen $\xrightarrow{\text{删除c}}$ sitthen $\xrightarrow{\text{删除h}}$ sitten $\xrightarrow{\text{删除e}}$ sittn $\xrightarrow{\text{插入i}}$ sittin $\xrightarrow{\text{插入g}}$ sitting

编辑操作示例

A = kittchen

↓ 编辑操作

B = sitting

操作名称	操作示例
删除	kittchen → kitchen
插入	kithen → kitchen
替换	kitchem → kitchen

6次

- 方案1 kittchen $\xrightarrow{k\to s}$ sittchen $\xrightarrow{\text{删除c}}$ sitthen $\xrightarrow{\text{删除h}}$ sitten $\xrightarrow{\text{删除e}}$ sittn $\xrightarrow{\text{插入i}}$ sittin $\xrightarrow{\text{插入g}}$ sitting

5次

- 方案2 kittchen $\xrightarrow{k\to s}$ sittchen $\xrightarrow{\text{删除c}}$ sitthen $\xrightarrow{\text{删除h}}$ sitten $\xrightarrow{e\to i}$ sittin $\xrightarrow{\text{插入g}}$ sitting

编辑操作示例

A = kittchen

编辑操作

B = sitting

操作名称	操作示例
删除	kittchen → kitchen
插入	kithen → kitchen
替换	kitchem → kitchen

- 方案1 kittchen $\xrightarrow{k \to s}$ sittchen $\xrightarrow{删除c}$ sitthen $\xrightarrow{删除h}$ sitten $\xrightarrow{删除e}$ sittn $\xrightarrow{插入i}$ sittin $\xrightarrow{插入g}$ sitting
- 方案2 kittchen $\xrightarrow{k \to s}$ sittchen $\xrightarrow{删除c}$ sitthen $\xrightarrow{删除h}$ sitten $\xrightarrow{e \to i}$ sittin $\xrightarrow{插入g}$ sitting

编辑距离：编辑操作次数

编辑操作示例

A = kittchen

↓ 编辑操作

B = sitting

操作名称	操作示例
删除	kittchen → kitchen
插入	kithen → kitchen
替换	kitchem → kitchen

- 方案1 kittchen $\xrightarrow{\text{k→s}}$ sittchen $\xrightarrow{\text{删除c}}$ sitthen $\xrightarrow{\text{删除h}}$ sitten $\xrightarrow{\text{删除e}}$ sittn $\xrightarrow{\text{插入i}}$ sittin $\xrightarrow{\text{插入g}}$ sitting
- 方案2 kittchen $\xrightarrow{\text{k→s}}$ sittchen $\xrightarrow{\text{删除c}}$ sitthen $\xrightarrow{\text{删除h}}$ sitten $\xrightarrow{\text{e→i}}$ sittin $\xrightarrow{\text{插入g}}$ sitting

问题：如何求出最少的编辑操作数（最小编辑距离）？

问题定义

编辑距离问题

Minimum Edit Distance, MED

输入

- 长度为n的字符串s，长度为m的字符串t

输出

- 求出一组编辑操作$O = < e_1, e_2, \ldots e_d >$，令

优化目标

$$\min |O|$$

约束条件

$$s.t.\ 字符串s经过O的操作后满足s = t$$

问题结构分析

- **给出问题表示**
  - $D[i,j]$：字符串$s[1..i]$变为$t[1..j]$的最小编辑距离

$$\boxed{s[1..i]} \underbrace{\xrightarrow{\qquad\qquad\qquad}}_{D[i,j]} \boxed{t[1..j]}$$

- **明确原始问题**
  - $D[n,m]$：字符串$s[1..n]$变为$t[1..m]$的最小编辑距离

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：回顾与启发

- 最长公共子序列
  - 如果$s_i \neq t_j$

s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

  - 如果$s_i = t_j$

s	A	B	C	B	D	A	B
t	B	D	C	A	B		

s	A	B	C	B	D	A	B
t	B	D	C	A	B		

考察末尾元素

递推关系建立

- 考察末尾元素
  - 删除

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

→ ?

  - 插入

s	A	B	C	B	D	A	B	?

t	B	D	C	A	B	A

→ ?

  - 替换

							?
s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

→ ?

问题结构分析
↓
递推关系建立
↓
自底向上计算
↓
最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：删除

s	*A*	*B*	*C*	*B*	*D*	*A*	*B*

t	*B*	*D*	*C*	*A*	*B*	*A*

→

s	*A*	*B*	*C*	*B*	*D*	*A*	*B*

t	*B*	*D*	*C*	*A*	*B*	*A*

问题结构分析

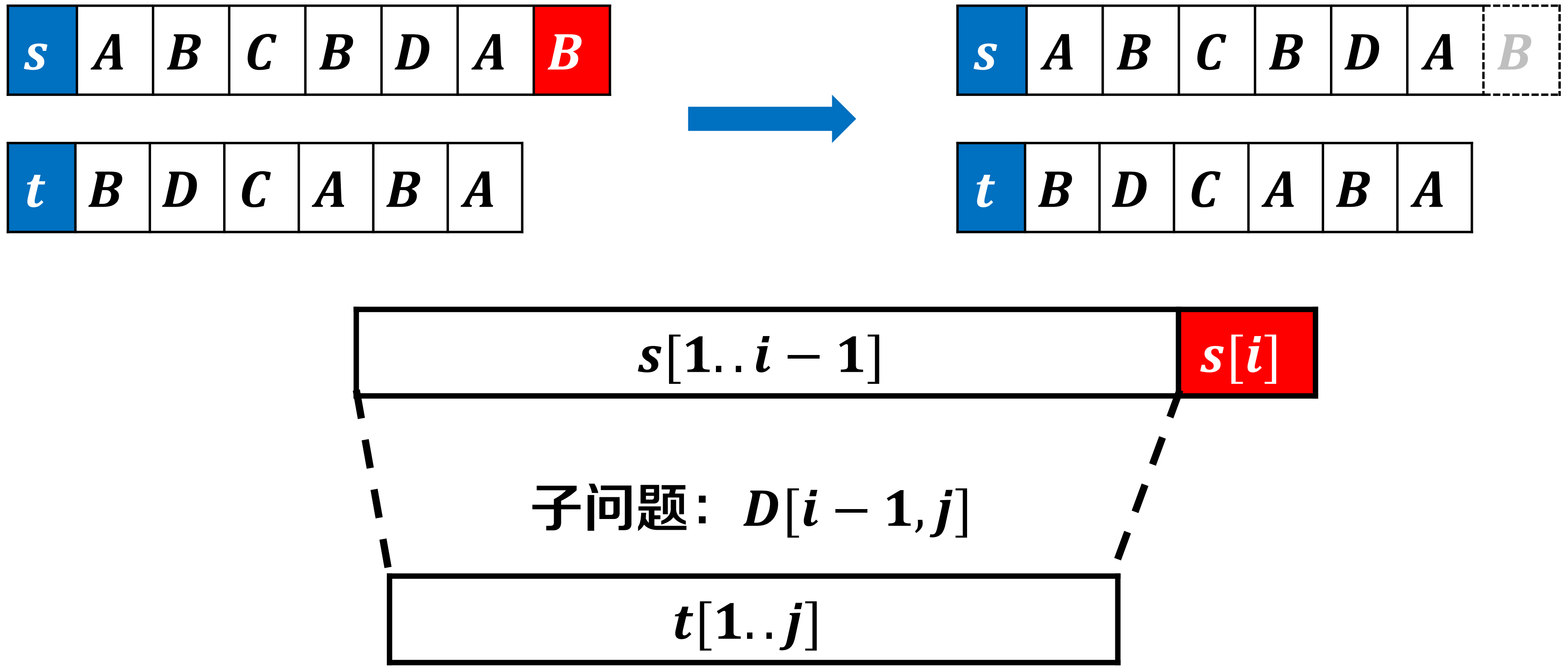
递推关系建立

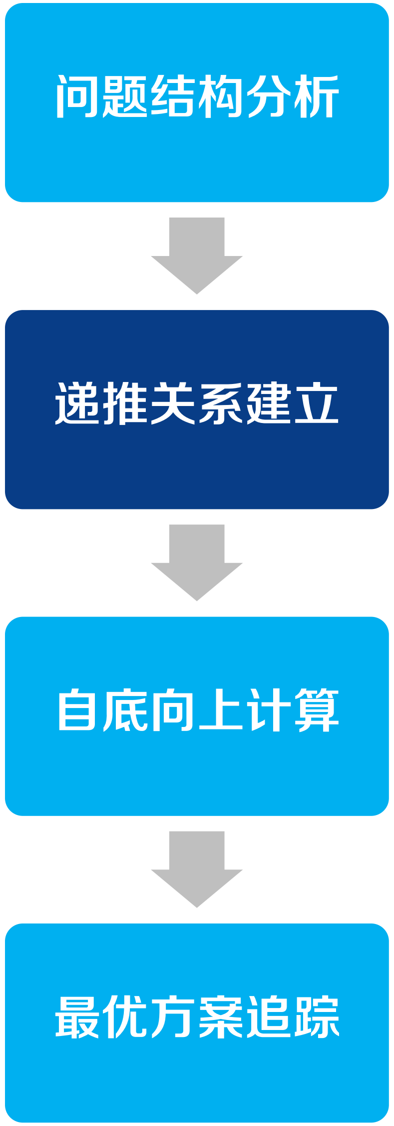
自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：删除





递推关系建立：分析最优（子）结构

- 考察末尾元素：删除

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

$s[1..i-1]$ $s[i]$

子问题：$D[i-1,j]$

$t[1..j]$

$$\boxed{s[1..i]} \underbrace{\xrightarrow{\text{删除}s[i]} \boxed{s[1..i-1]} \xrightarrow{D[i-1,j]}}_{D[i,j]} \boxed{t[1..j]}$$

问题结构分析

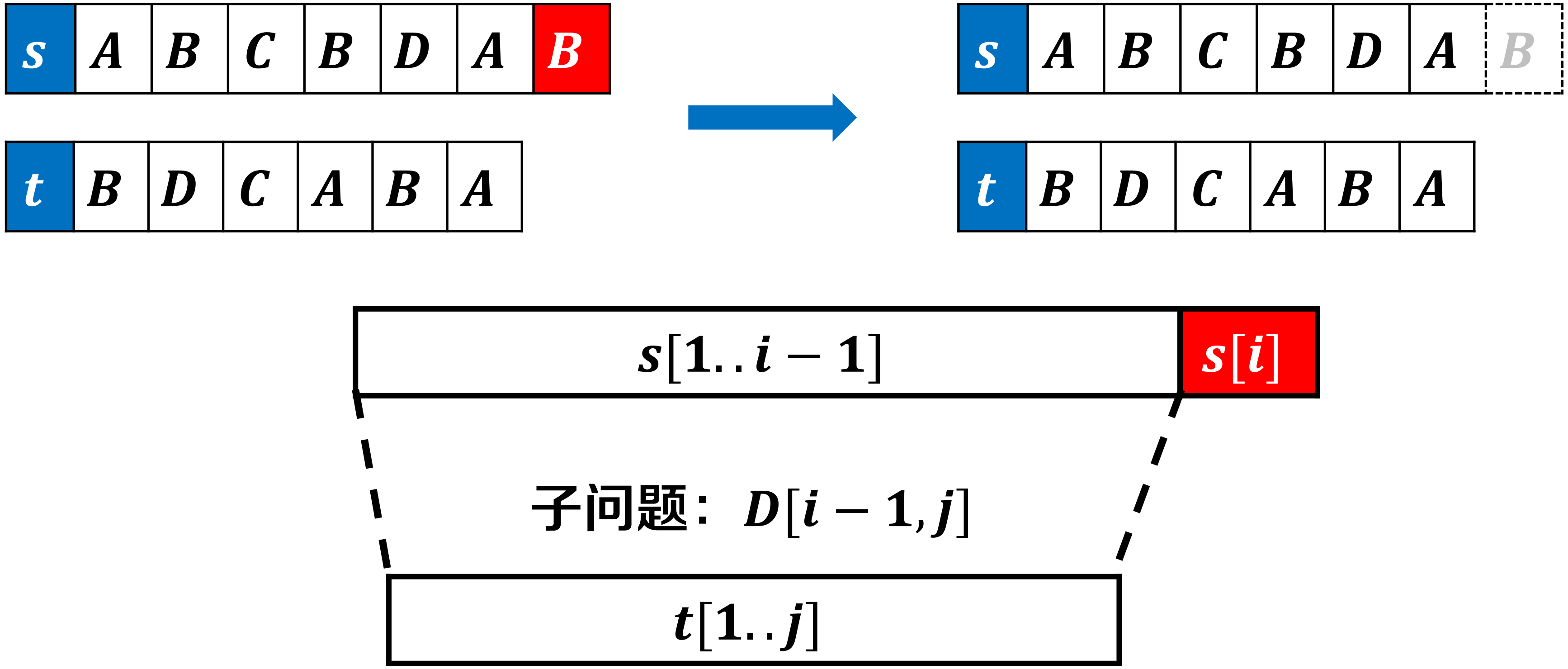
递推关系建立

自底向上计算

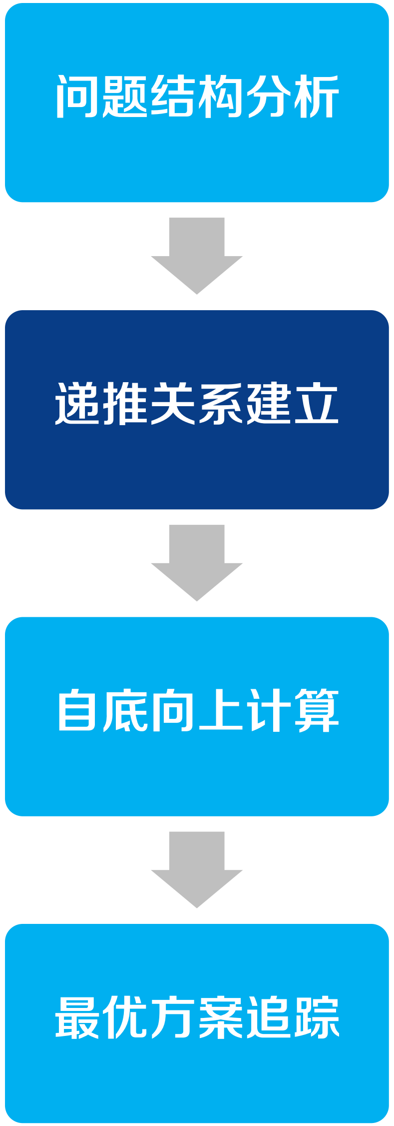
最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：删除



- $D[i,j] = D[i-1,j] + 1$



递推关系建立：分析最优（子）结构

- 考察末尾元素：删除

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

$s[1..i-1]$	$s[i]$

子问题：$D[i-1,j]$

$t[1..j]$

- $\boxed{D[i,j]} = \boxed{D[i-1,j]} + 1$

最优子结构

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

s	A	B	C	B	D	A	B	?

t	B	D	C	A	B	A

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

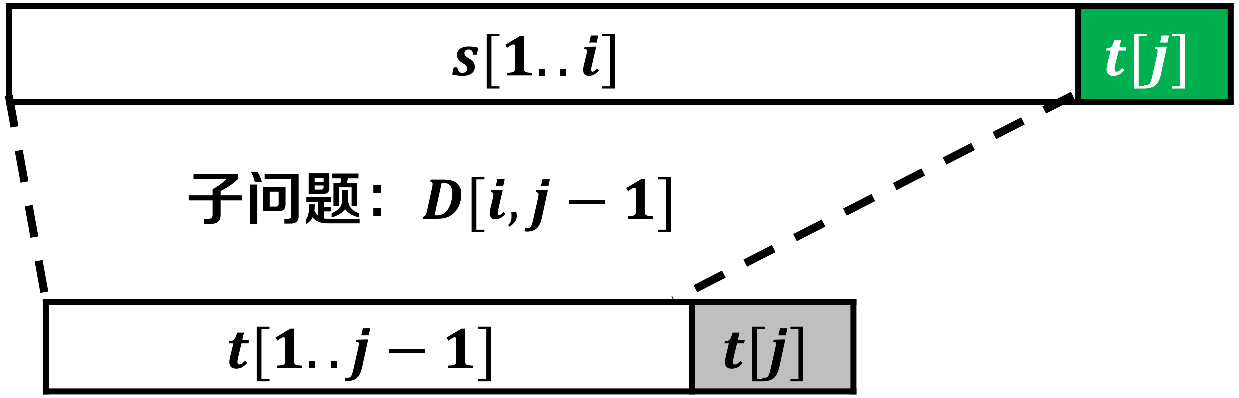
s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A



问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A

$s[1..i]$	$t[j]$

子问题：$D[i,j-1]$

$t[1..j-1]$	$t[j]$

$$\underbrace{s[1..i] \xrightarrow{D[i,j-1]} t[1..j-1] \xrightarrow{\text{插入}t[j]}}_{D[i,j]} t[1..j]$$

- 问题结构分析
- **递推关系建立**
- 自底向上计算
- 最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

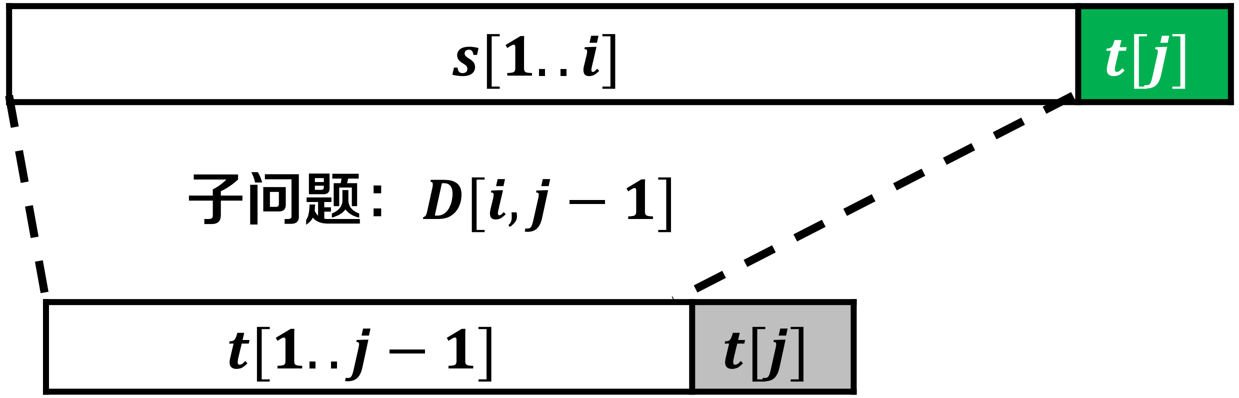
s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	B	A

t	B	D	C	A	B	A



- $D[i,j] = D[i,j-1] + 1$

1. 问题结构分析
2. 递推关系建立
3. 自底向上计算
4. 最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：插入

s	A	B	C	B	D	A	B	A
t	B	D	C	A	B	A		

s	A	B	C	B	D	A	B	A
t	B	D	C	A	B	A		

$s[1..i]$ $t[j]$

子问题：$D[i,j-1]$

$t[1..j-1]$ $t[j]$

- $D[i,j] = D[i,j-1] + 1$

最优子结构

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：替换

s	*A*	*B*	*C*	*B*	*D*	*A*	*B*

t	*B*	*D*	*C*	*A*	*B*	*A*

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：替换

s	A	B	C	B	D	A	?

t	B	D	C	A	B	A

- 问题结构分析
- **递推关系建立**
- 自底向上计算
- 最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：替换

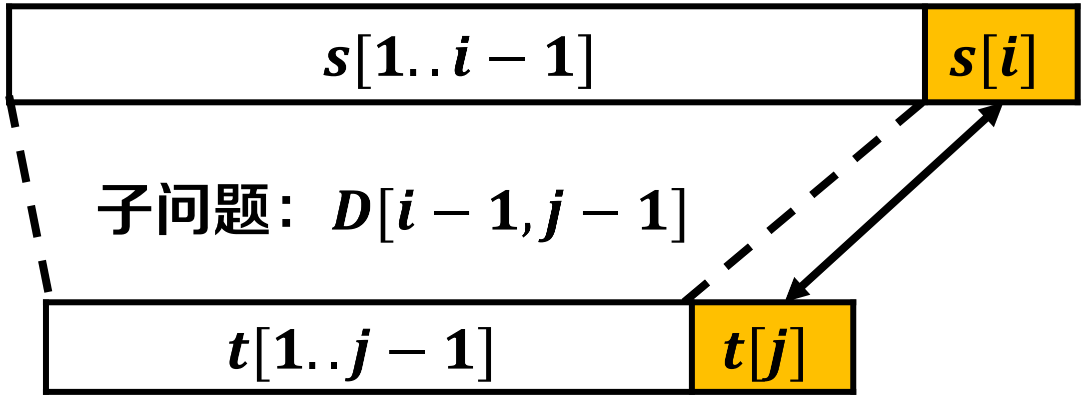
s	A	B	C	B	D	A	A

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	A

t	B	D	C	A	B	A



问题结构分析

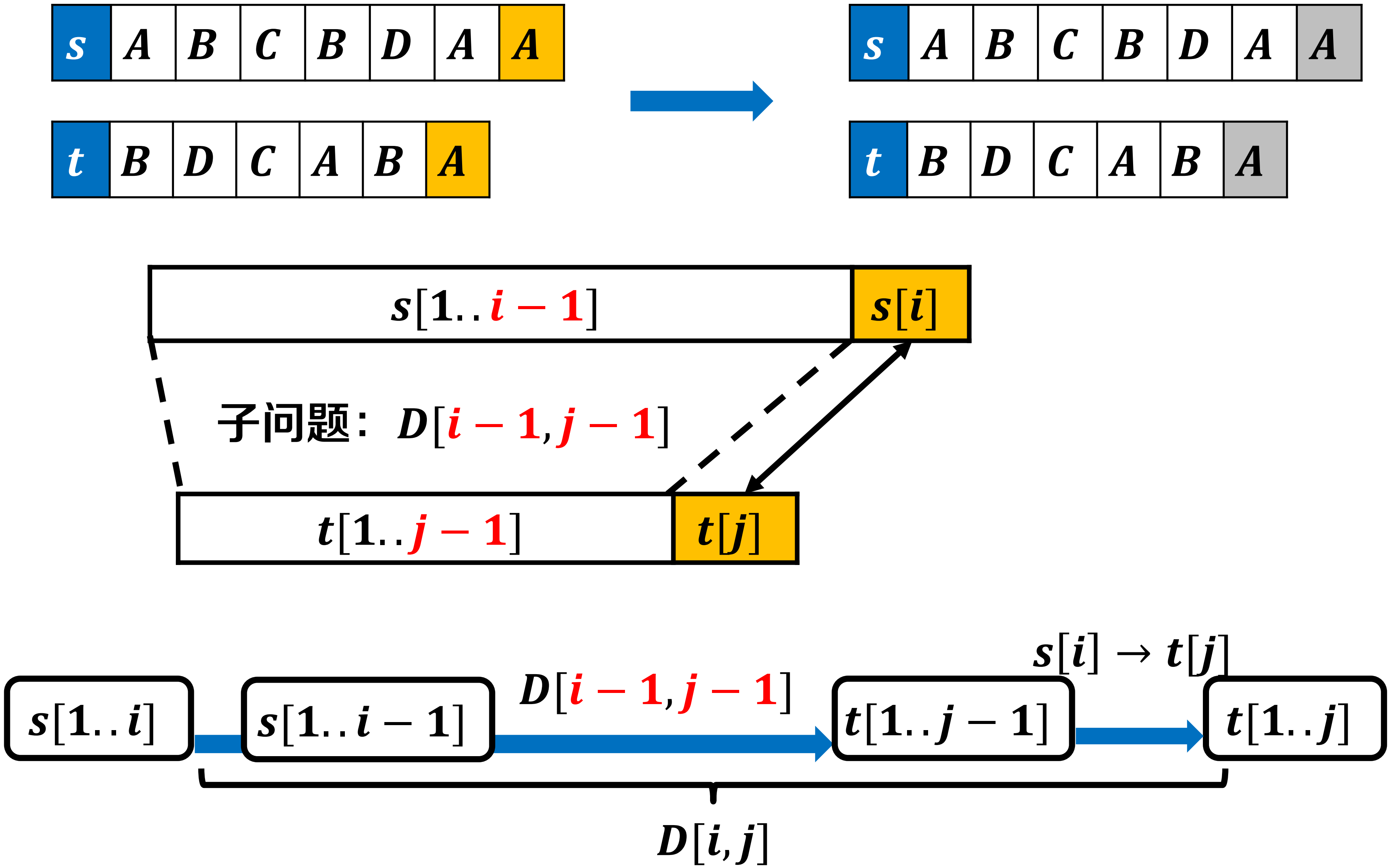
递推关系建立

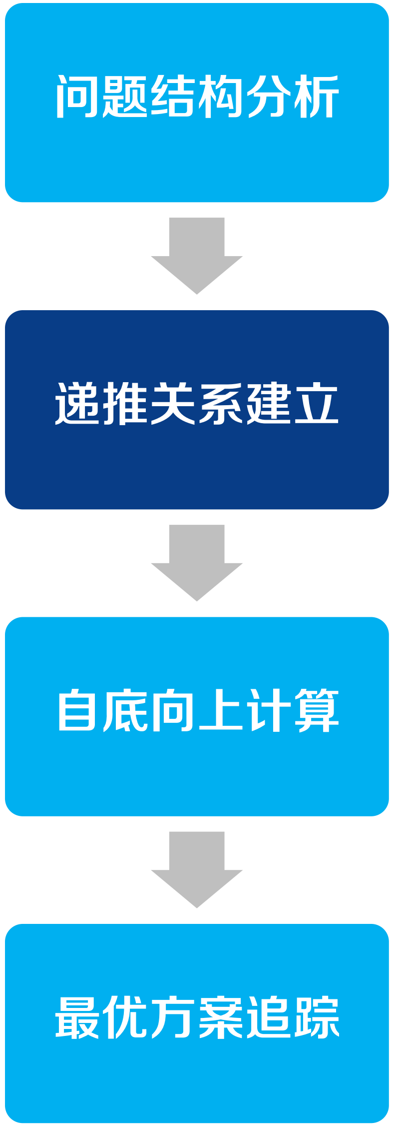
自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

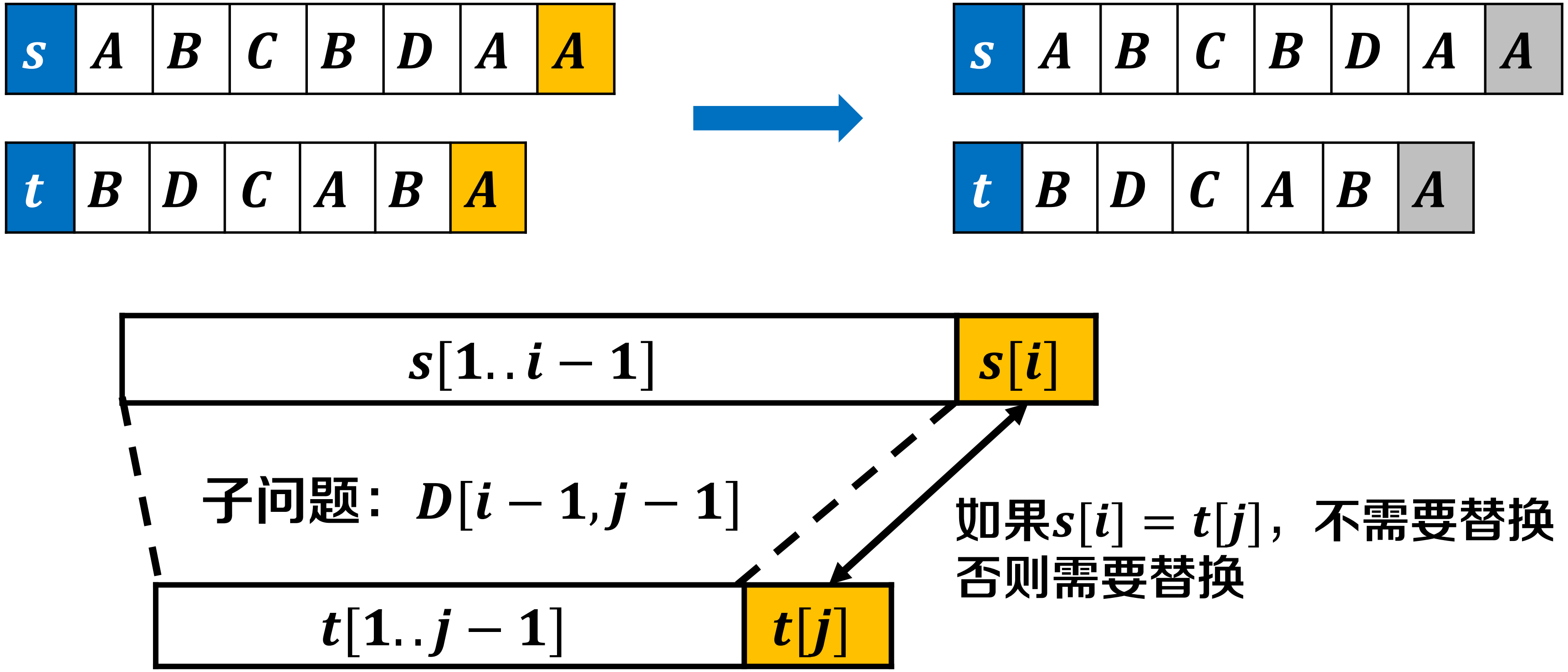
- 考察末尾元素：替换





递推关系建立：分析最优（子）结构

- 考察末尾元素：替换



如果$s[i] = t[j]$，不需要替换
否则需要替换

- $D[i,j] = D[i-1,j-1] + \begin{cases} 0, & if\ s[i] = t[j] \\ 1, & if\ s[i] \neq t[j] \end{cases}$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 考察末尾元素：替换

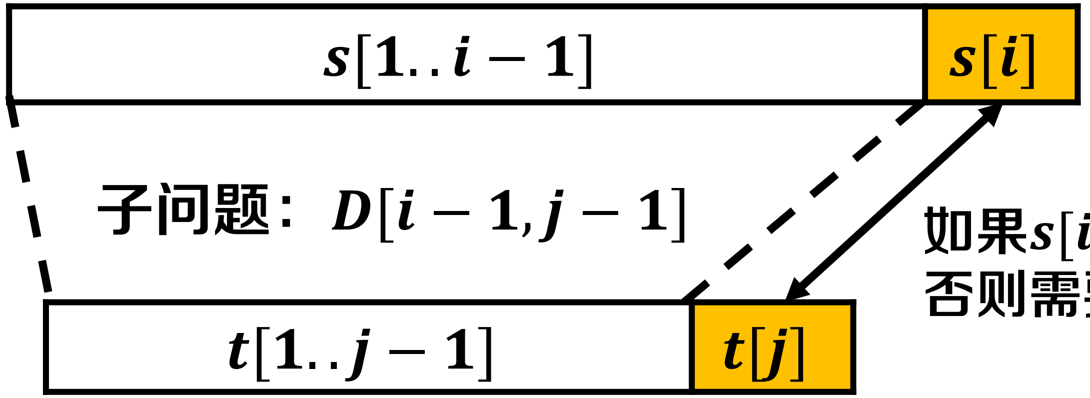
s	A	B	C	B	D	A	A

t	B	D	C	A	B	A

→

s	A	B	C	B	D	A	A

t	B	D	C	A	B	A



如果$s[i]=t[j]$，不需要替换
否则需要替换

- $$\boxed{D[i,j]}=\boxed{D[i-1,j-1]}+\begin{cases}0, & if\ s[i]=t[j]\\1, & if\ s[i]\neq t[j]\end{cases}$$

最优子结构

问题结构分析
递推关系建立
自底向上计算
最优方案追踪

递推关系建立：构造递推公式

- 综合上面三种方式
  - $$D[i,j]=\min\begin{cases} D[i-1,j]+1 & \text{删除} \\ D[i,j-1]+1 & \text{插入} \\ D[i-1,j-1]+\begin{cases}0, & if\ s[i]=t[j]\\ 1, & if\ s[i]\neq t[j]\end{cases} & \text{替换}\end{cases}$$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：构造递推公式

- 最小编辑距离 vs. 最长公共子序列

  - $$D[i,j]=\min\begin{cases} D[i-1,j]+1 & \text{删除} \\ D[i,j-1]+1 & \text{插入} \\ D[i-1,j-1]+\begin{cases}0, & if\ s[i]=t[j]\\ 1, & if\ s[i]\neq t[j]\end{cases} & \text{替换}\end{cases}$$

  - $$C[i,j]=\begin{cases}\max\{C[i-1,j],C[i,j-1]\}, & x_i\neq y_j\\ C[i-1,j-1]+1 & ,x_i=y_j\end{cases}$$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $D[i,0]=i$
    - 把长度为i的串变为空串至少需要i次操作（删除）
  - $D[0,j]=j$
    - 把空串变为长度为j的串至少需要j次操作（插入）

$i \backslash j$	0	1	2	…	$m-1$	m
0	0	1	2	…	$m-1$	m
1	1					
2	2					
…	…					
$n-1$	$n-1$					
n	n					

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 递推公式

  - $$D[i,j]=\min\begin{cases} D[i-1,j]+1 & \text{删除} \\ D[i,j-1]+1 & \text{插入} \\ D[i-1,j-1]+\begin{cases}0, & if\ s[i]=t[j] \\ 1, & if\ s[i]\neq t[j]\end{cases} & \text{替换}\end{cases}$$

$i \backslash j$	0	1	2	$\dots$	$m-1$	m
0	0	1	2	$\dots$	$m-1$	m
1	1	$D[i-1,j-1]+\begin{cases}0\\1\end{cases}$				
2	2			$D[i-1,j]+1$		
$\dots$	$\dots$			$\rightarrow D[i,j]$		
$n-1$	$n-1$		$D[i,j-1]+1$			
n	n					

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：依次计算问题

- 递推公式

$$
D[i,j]=\min\begin{cases} D[i-1,j]+1 & \text{删除} \\ D[i,j-1]+1 & \text{插入} \\ D[i-1,j-1]+\begin{cases}0, & if\ s[i]=t[j]\\ 1, & if\ s[i]\neq t[j]\end{cases} & \text{替换}\end{cases}
$$

$i \backslash j$	0	1	2	…	$m-1$	m
0	0	1	2	…	$m-1$	m
1	1					
2	2					
…	…					
$n-1$	$n-1$					
n	n					

问题结构分析 → 递推关系建立 → 自底向上计算 → 最优方案追踪

最优方案追踪：记录决策过程

- 追踪数组Rec，记录子问题来源

$s[1..i-1]$ $s[i]$

子问题：$D[i-1,j]$

$t[1..j]$

$D[i-1,j-1]$	$D[i-1,j]$
$D[i,j-1]$	$D[i,j]$

$Rec[i,j]$	子问题来源	操作
U	上侧，即$D[i-1,j]$	删除$s[i]$

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪：记录决策过程

- 追踪数组Rec，记录子问题来源

$s[1..i-1]$	$s[i]$	$t[j]$

子问题：$D[i,j-1]$

$t[1..j-1]$	$t[j]$

$D[i-1,j-1]$	$D[i-1,j]$
$D[i,j-1]$ $\longleftarrow$	$D[i,j]$

$Rec[i,j]$	子问题来源	操作
U	上侧，即$D[i-1,j]$	删除$s[i]$
L	左侧，即$D[i,j-1]$	插入$t[j]$

问题结构分析

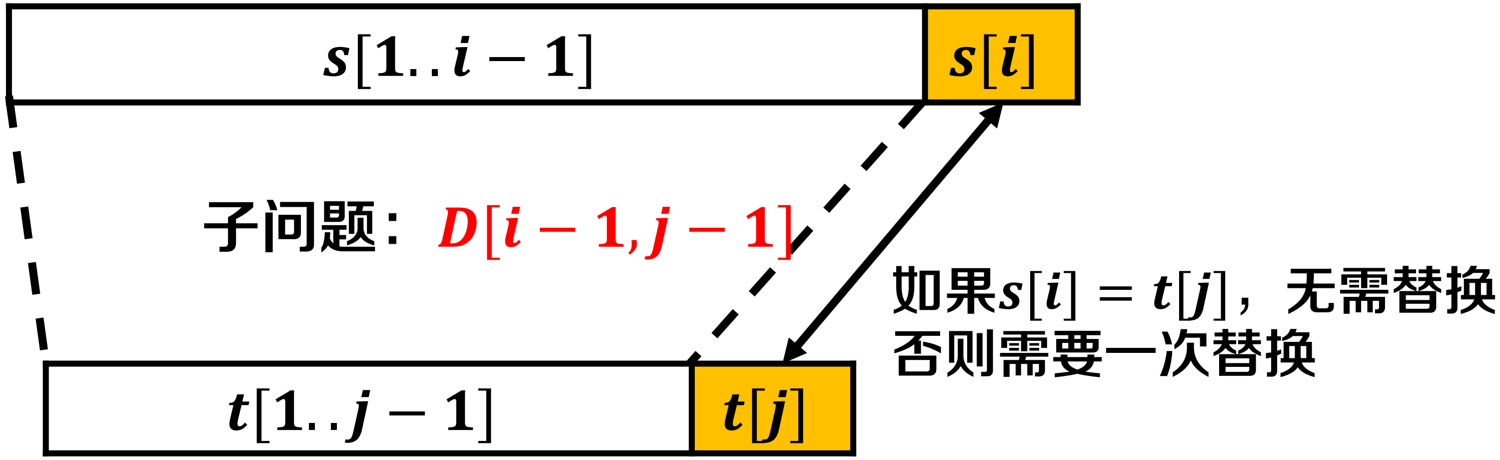
递推关系建立

自底向上计算

最优方案追踪

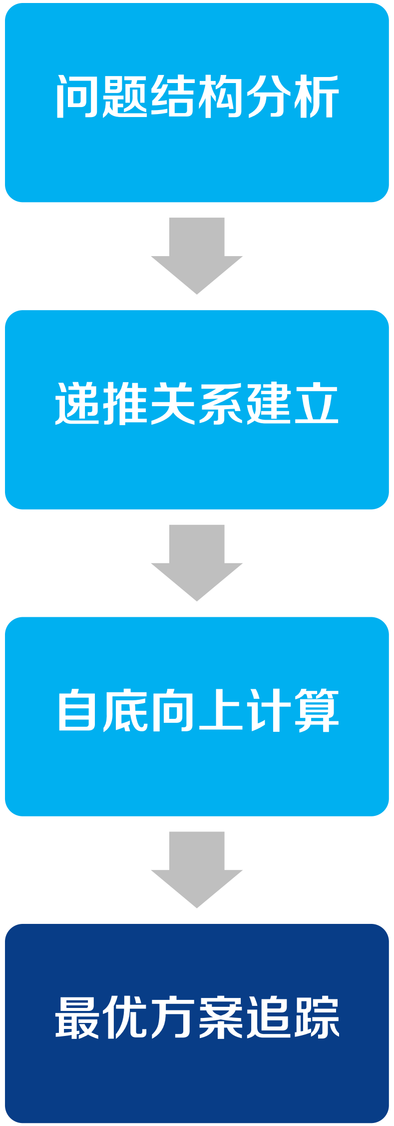
最优方案追踪：记录决策过程

- 追踪数组Rec，记录子问题来源



$D[i-1,j-1]$	$D[i-1,j]$
$D[i,j-1]$	$D[i,j]$

$Rec[i,j]$	子问题来源	操作
U	上侧，即$D[i-1,j]$	删除$s[i]$
L	左侧，即$D[i,j-1]$	插入$t[j]$
LU	左上，即$D[i-1,j-1]$	用$t[j]$替换$s[i]$/无操作



最优方案追踪：输出最优方案

- 根据数组Rec，输出最少编辑操作

$i \backslash j$	0	1	2	…	m
0					
1					
…					
n					

$Rec[i,j] = L$

$Rec[i,j] = LU$

$Rec[i,j] = U$

$Rec[i,j]$	子问题来源	操作
U	上侧，即$D[i-1,j]$	删除$s[i]$
L	左侧，即$D[i,j-1]$	插入$t[j]$
LU	左上，即$D[i-1,j-1]$	用$t[j]$替换$s[i]$/无操作

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	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						

初始化

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j] + 1 & \text{删除A} \\ D[i,j-1] + 1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

$D[i-1,j]$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j]=\min\begin{cases} D[i-1,j]+1 & \text{删除A} \\ D[i,j-1]+1 & \text{插入B} \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j]\\ 1, if\ s[i]\neq t[j]\end{cases} & \end{cases}$$

D

$D[i-1,j]$

$i \backslash j$	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						

$D[i,j-1]$

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	[illegible]	[illegible]	A	

$s[i] \neq t[j]$

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 & \text{删除A} \\ D[i,j-1]+1 & \text{插入B} \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} & \end{cases}$$

A替换为B

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						

$D[i-1,j]$

$D[i-1,j-1]$

$D[i,j-1]$

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$s[i] \neq t[j]$

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

$D[i-1,j]$

$D[i-1,j-1]$

$D[i,j-1]$

D

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

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$s[i] \neq t[j]$

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

$D[i-1,j]$

$D[i-1,j-1]$

$D[i,j-1]$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$s[i] = t[j]$

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

无需替换

$D[i-1,j]$

$D[i-1,j-1]$

$D[i,j-1]$

D

i \ *j*	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	1	2	3	3		
2	2						
3	3						
4	4						
5	5						
6	6						
7	7						

c

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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	4						
5	5						
6	6						
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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	3						
4	4						
5	5						
6	6						
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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	
3	3						
4	4						
5	5						
6	6						
7	7						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j]\end{cases}\end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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				
5	5						
6	6						
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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			
5	5						
6	6						
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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	
5	5						
6	6						
7	7						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j]=\min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j]\\ 1, if\ s[i]\neq t[j]\end{cases}\end{cases}$$

D

i \ 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			
6	6						
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j]=\min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j]\\ 1, if\ s[i]\neq t[j]\end{cases}\end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases} 0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j]\end{cases}\end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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				
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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			
7	7						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min\begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1]+\begin{cases}0, if\ s[i]=t[j] \\ 1, if\ s[i]\neq t[j]\end{cases}\end{cases}$$

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						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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	
7	7						

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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						

Rec

i \ j	0	1	2	3	4	5	6
0		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						

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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					

Rec

i \ j	0	1	2	3	4	5	6
0		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					

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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				

Rec

i \ j	0	1	2	3	4	5	6
0		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				

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

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			

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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		

Rec

i \ j	0	1	2	3	4	5	6
0		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		

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ 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	

Rec

i \ j	0	1	2	3	4	5	6
0		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	

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

D

i \ *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

Rec

i \ *j*	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

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	[illegible]	[illegible]	[illegible]	4
6	6	5	4	4	4	5	4
7	7	6	5	5	5	4	5

最优解

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

Rec

i \ j	0	1	2	3	4	5	6
0		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

操作:

插入A

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作：

无需操作
插入A

Rec

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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:

无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:

用C替换D
无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:

用D替换B
用C替换D
无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:

删除C
用D替换B
用C替换D
无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:

无需操作
删除C
用D替换B
用C替换D
无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

算法实例

	1	2	3	4	5	6	7
s	A	B	C	B	D	A	B
t	B	D	C	A	B	A	

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

操作:
删除A
无需操作
删除C
用D替换B
用C替换D
无需操作
无需操作
插入A

Rec

i \ j	0	1	2	3	4	5	6
0		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

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

输入: 字符串s和t
输出: s和t的最小编辑距离
$n \leftarrow \text{length}(s)$
$m \leftarrow \text{length}(t)$
新建$D[0..n, 0..m]$, $Rec[0..n, 0..m]$两个二维数组
//初始化
for $i \leftarrow 0$ *to* n **do**
    $D[i, 0] \leftarrow i$
    $Rec[i, 0] \leftarrow$ "U"
end
for $j \leftarrow 0$ *to* m **do**
    $D[0, j] \leftarrow j$
    $Rec[0, j] \leftarrow$ "L"
end

初始化

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i−1, j−1] + c
        delete ← D[i−1, j] + 1
        insert ← D[i, j−1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i−1, j−1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j−1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i−1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

依次计算子问题

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i − 1, j − 1] + c
        delete ← D[i − 1, j] + 1
        insert ← D[i, j − 1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i − 1, j − 1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j − 1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i − 1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

替换/无需操作

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i − 1, j − 1] + c
        delete ← D[i − 1, j] + 1
        insert ← D[i, j − 1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i − 1, j − 1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j − 1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i − 1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

采用替换操作

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i−1, j−1] + c
        delete ← D[i−1, j] + 1
        insert ← D[i, j−1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i−1, j−1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j−1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i−1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

记录编辑距离和操作

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i − 1, j − 1] + c
        delete ← D[i − 1, j] + 1
        insert ← D[i, j − 1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i − 1, j − 1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j − 1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i − 1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

采取插入操作

动态规划算法：伪代码

- **Minimum-Edit-Distance(s, t)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i − 1, j − 1] + c
        delete ← D[i − 1, j] + 1
        insert ← D[i, j − 1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i − 1, j − 1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j − 1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i − 1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

采取删除操作

最优方案追踪：伪代码

- **Print-MED(Rec, s, t, i, j)**

```
输入: 矩阵Rec, 字符串s, t, 位置索引i和j
输出: 操作序列
if i = 0 and j = 0 then
  | return NULL
end
if Rec[i, j] = "LU" then
  | Print-MED(Rec, s, t, i − 1, j − 1)
  | if s_i = t_j then
  |   | print "无需操作"
  | end
  | else
  |   | print "用t_j替换s_i"
  | end
end
else if Rec[i, j] = "U" then
  | Print-MED(Rec, s, t, i − 1, j)
  | print "删除s_i"
end
else
  | Print-MED(Rec, s, t, i, j − 1)
  | print "插入t_j"
end
```

$\boxed{\text{采取替换操作}}$

最优方案追踪：伪代码

- Print-MED(Rec, s, t, i, j)

```
输入: 矩阵Rec, 字符串s,t, 位置索引i和j
输出: 操作序列
if i = 0 and j = 0 then
 | return NULL
end
if Rec[i,j] = "LU" then
 | Print-MED(Rec, s, t, i − 1, j − 1)
 | if s_i = t_j then
 |  | print "无需操作"
 | end
 | else
 |  | print "用t_j替换s_i"
 | end
end
else if Rec[i,j] = "U" then
 | Print-MED(Rec, s, t, i − 1, j)
 | print "删除s_i"
end
else
 | Print-MED(Rec, s, t, i, j − 1)
 | print "插入t_j"
end
```

递归输出子问题方案

最优方案追踪：伪代码

- **Print-MED(Rec, s, t, i, j)**

```
输入: 矩阵Rec, 字符串s,t, 位置索引i和j
输出: 操作序列
if i = 0 and j = 0 then
 | return NULL
end
if Rec[i,j] = "LU" then
 | Print-MED(Rec, s, t, i-1, j-1)
 | if s_i = t_j then
 |  | print "无需操作"
 | end
 | else
 |  | print "用t_j替换s_i"
 | end
end
else if Rec[i,j] = "U" then
 | Print-MED(Rec, s, t, i-1, j)
 | print "删除s_i"
end
else
 | Print-MED(Rec, s, t, i, j-1)
 | print "插入t_j"
end
```

替换/无操作

最优方案追踪：伪代码

- **Print-MED(Rec, s, t, i, j)**

```
输入: 矩阵Rec, 字符串s, t, 位置索引i和j
输出: 操作序列
if i = 0 and j = 0 then
    return NULL
end
if Rec[i, j] = "LU" then
    Print-MED(Rec, s, t, i − 1, j − 1)
    if s_i = t_j then
        print "无需操作"
    end
    else
        print "用t_j替换s_i"
    end
end
else if Rec[i, j] = "U" then
    Print-MED(Rec, s, t, i − 1, j)
    print "删除s_i"
end
else
    Print-MED(Rec, s, t, i, j − 1)
    print "插入t_j"
end
```

采取删除操作

最优方案追踪：伪代码

- **Print-MED**$(\boldsymbol{Rec, s, t, i, j})$

输入: 矩阵Rec, 字符串s, t, 位置索引i和j

输出: 操作序列

if $i = 0$ and $j = 0$ **then**
  return $NULL$
end
if $Rec[i, j] =$ "LU" **then**
  Print-MED$(Rec, s, t, i-1, j-1)$
  if $s_i = t_j$ **then**
    print "**无需操作**"
  end
  else
    print "**用**$\boldsymbol{t_j}$**替换**$\boldsymbol{s_i}$"
  end
end
else if $Rec[i, j] =$ "U" **then**
  Print-MED$(Rec, s, t, i-1, j)$
  print "**删除**$\boldsymbol{s_i}$"
end
else
  Print-MED$(Rec, s, t, i, j-1)$
  print "**插入**$\boldsymbol{t_j}$"
end

采取插入操作

时间复杂度分析

//动态规划

```
for i ← 1 to n do
    for j ← 1 to m do
        c ← 0
        if s_i ≠ t_j then
            c ← 1
        end
        replace ← D[i − 1, j − 1] + c
        delete ← D[i − 1, j] + 1
        insert ← D[i, j − 1] + 1
        if replace = min{delete, insert, replace} then
            D[i, j] ← D[i − 1, j − 1] + c
            Rec[i, j] ← "LU"
        end
        else if insert = min{delete, insert, replace} then
            D[i, j] ← D[i, j − 1] + 1
            Rec[i, j] ← "L"
        end
        else
            D[i, j] ← D[i − 1, j] + 1
            Rec[i, j] ← "U"
        end
    end
end
```

$O(m)$ $O(mn)$

时间复杂度： $O(mn)$

小结

最长公共子序列

如果$s_i \neq t_j$

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

如果$s_i = t_j$

s	A	B	C	B	D	A	B

t	B	D	C	A	B

$$C[i,j] = \begin{cases} \max\{C[i-1,j], C[i,j-1]\}, x_i \neq y_j \\ C[i-1,j-1]+1 \qquad\qquad\quad , x_i = y_j \end{cases}$$

最小编辑距离

删除

s	A	B	C	B	D	A	B

t	B	D	C	A	B	A

插入

s	A	B	C	B	D	A	B	?

t	B	D	C	A	B	A

替换

s	A	B	C	B	D	A	B	?

t	B	D	C	A	B	A

$$D[i,j] = \min \begin{cases} D[i-1,j]+1 \\ D[i,j-1]+1 \\ D[i-1,j-1] + \begin{cases} 0, if\ s[i] = t[j] \\ 1, if\ s[i] \neq t[j] \end{cases} \end{cases}$$

动态规划篇：最长公共子串问题

问题背景

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果

- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

问题背景

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

X的子序列

X_{seq}	A	C	B	B

问题背景

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 子串
  - 给定序列中零个或多个**连续**的元素（如字符）组成的子序列
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

X的子序列

X_{seq}	A	C	B	B

问题背景

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 子串
  - 给定序列中零个或多个**连续**的元素（如字符）组成的子序列
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

X的子序列

X_{seq}	A	C	B	B

X的子串

X_{str}	A	B	C	B

问题背景

- 子序列
  - 将给定序列中零个或多个元素（如字符）去掉后所得结果
- 子串
  - 给定序列中零个或多个**连续**的元素（如字符）组成的子序列
- 示例
  - 给定序列X

X	A	B	C	B	D	A	B

X的子序列

X_{seq}	A	C	B	B

X的子串

X_{str}	B	C	B

问题背景：公共子串

- 给定两个序列X和Y

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 公共子串示例

问题背景：公共子串

- 给定两个序列X和Y

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 公共子串示例

X_1	B

Y_1	B

问题背景：公共子串

- 给定两个序列X和Y

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 公共子串示例

X_1	B

Y_1	B

X_2	B	C

Y_2	B	C

问题背景：公共子串

- 给定两个序列X和Y

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 公共子串示例

X_1	B

Y_1	B

X_2	B	C

Y_2	B	C

X_3	D	B	B

Y_3	D	B	B

问题背景：公共子串

- 给定两个序列X和Y

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 公共子串示例

X_1	B

Y_1	B

X_2	B	C

Y_2	B	C

X_3	D	B	B

Y_3	D	B	B

问题：如何求两个给定序列的最长公共子串？

问题定义

- 形式化定义

最长公共子串问题

Longest Common Substring Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

问题定义

- 形式化定义

最长公共子串问题

Longest Common Substring Problem

输入

- 序列$X =< x_1, x_2, \dots, x_n >$和序列$Y =< y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子串$Z =< z_1, z_2, \dots, z_l >$，令

问题定义

- 形式化定义

最长公共子串问题

Longest Common Substring Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子串$Z = < z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

$$s.t.\ Z = < x_i, x_{i+1}, \dots, x_{i+l-1} > = < y_j, y_{j+1}, \dots, y_{j+l-1} >$$
$$(1 \le i \le n - l + 1; 1 \le j \le m - l + 1)$$

问题定义

- 形式化定义

最长公共子串问题

Longest Common Substring Problem

输入

- 序列$X=<x_1,x_2,\dots,x_n>$和序列$Y=<y_1,y_2,\dots,y_m>$

输出

- 求解一个公共子串$Z=<z_1,z_2,\dots,z_l>$，令

$$\max|Z|$$

优化目标

$$s.t.\ Z=<x_i,x_{i+1},\dots,x_{i+l-1}>=<y_j,y_{j+1},\dots,y_{j+l-1}>$$
$$(1\le i\le n-l+1;1\le j\le m-l+1)$$

形式化定义

最长公共子串问题

Longest Common Substring Problem

输入

- 序列$X = < x_1, x_2, \dots, x_n >$和序列$Y = < y_1, y_2, \dots, y_m >$

输出

- 求解一个公共子串$Z = < z_1, z_2, \dots, z_l >$，令

$$\max |Z|$$

优化目标

$$s.t.\ Z = < x_i, x_{i+1}, \dots, x_{i+l-1} > = < y_j, y_{j+1}, \dots, y_{j+l-1} >$$

$$(1 \le i \le n - l + 1; 1 \le j \le m - l + 1)$$

约束条件

蛮力枚举

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 序列X和序列Y各选择一个位置$X[i]$和$Y[j]$

蛮力枚举

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$

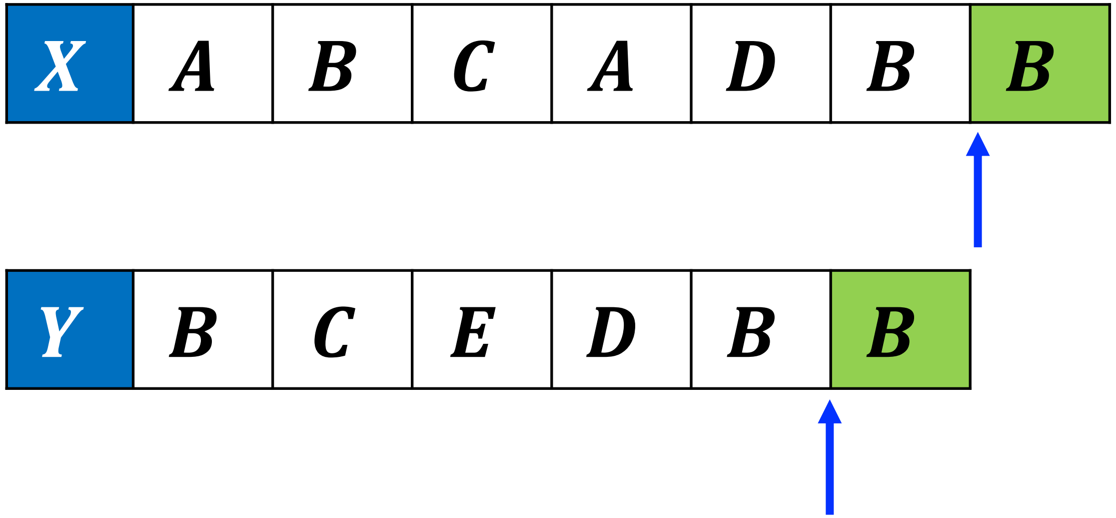
蛮力枚举

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

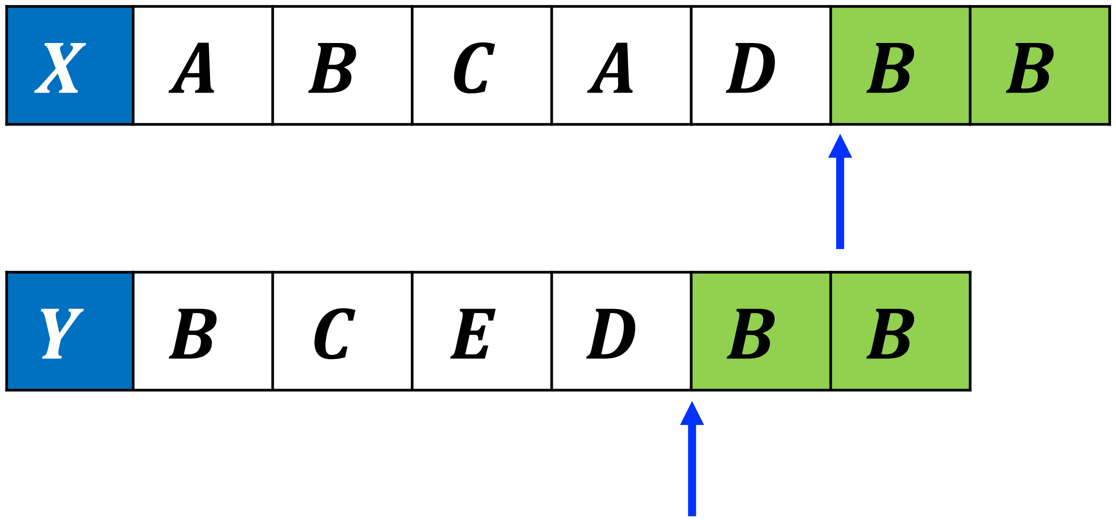
- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$
- 依次检查元素是否匹配

蛮力枚举



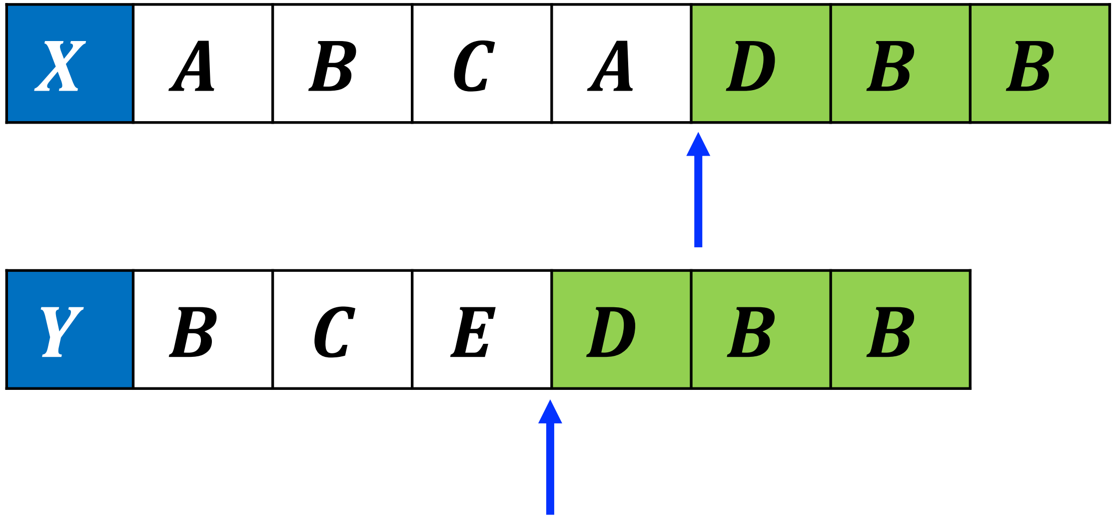
- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$
- 依次检查元素是否匹配
  - 元素相等继续匹配

蛮力枚举



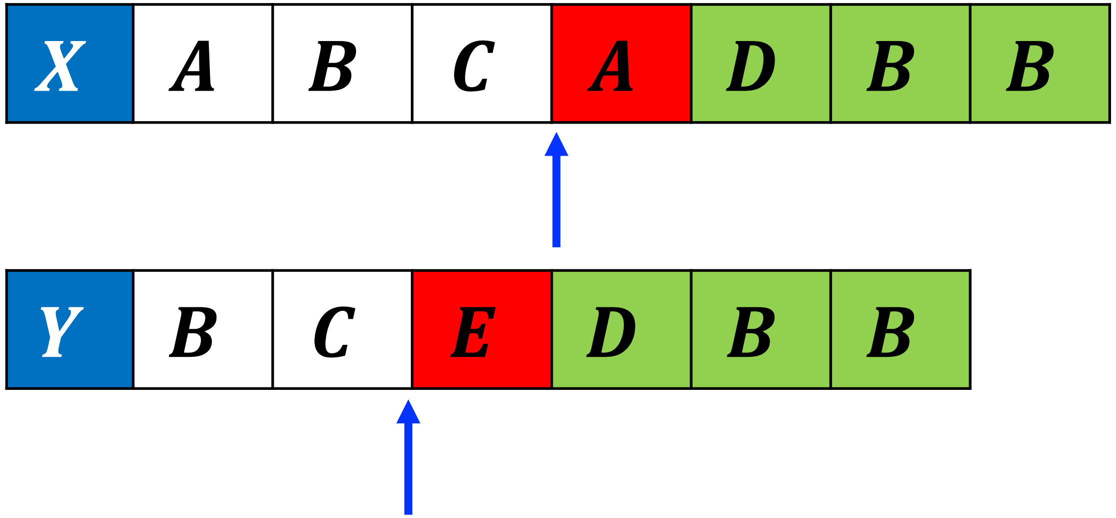
- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$
- 依次检查元素是否匹配
  - 元素相等继续匹配

蛮力枚举



- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$
- 依次检查元素是否匹配
  - 元素相等继续匹配

蛮力枚举



- 序列X和序列Y各选择一个位置$X[7]$和$Y[6]$
- 依次检查元素是否匹配
  - 元素相等继续匹配
  - 元素不等(或某序列已达端点)匹配终止

蛮力枚举

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

- 枚举所有的$X[i], Y[j]$
- 求以其为结尾的尽可能长的公共子串

蛮力枚举

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	B

最长公共子串长度为3

- 枚举所有的$X[i], Y[j]$
- 求以其为结尾的尽可能长的公共子串
- 记录最长公共子串长度

枚举观察

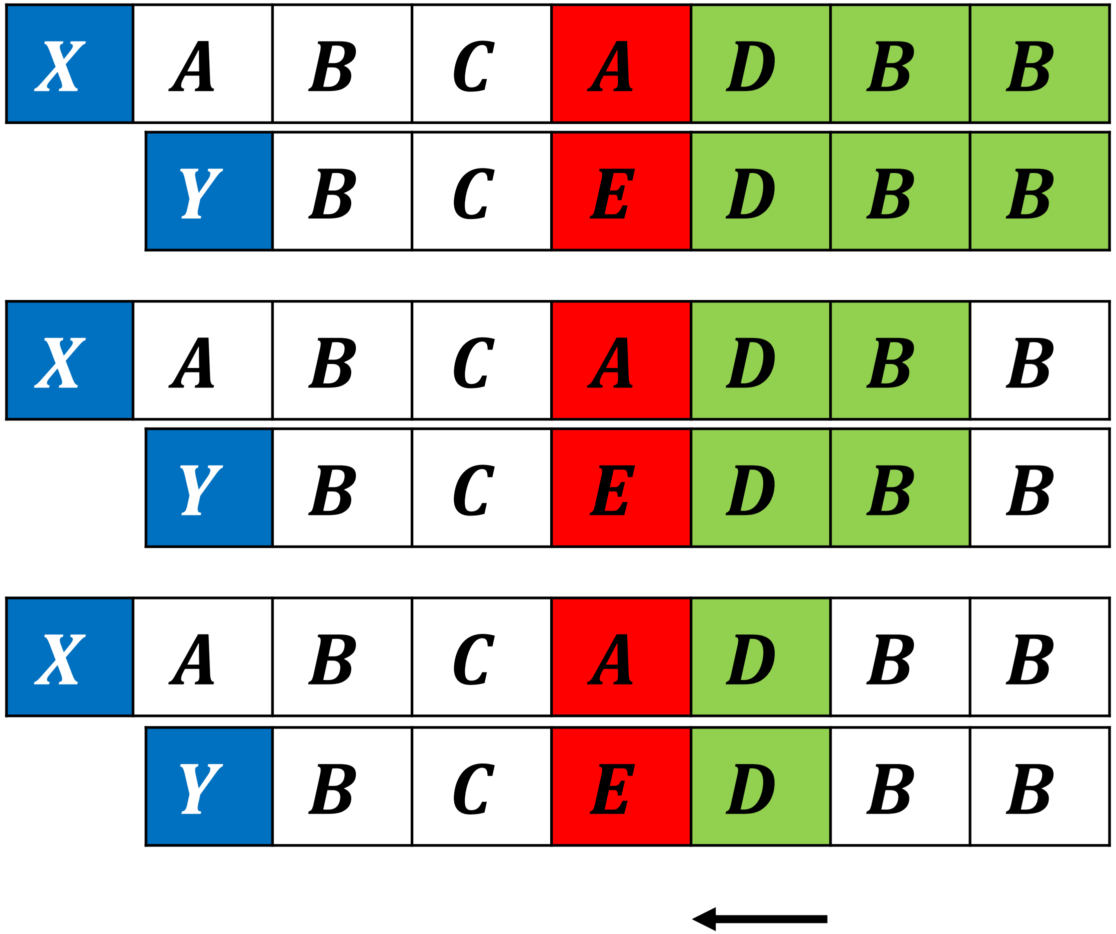
X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

枚举观察

X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

枚举观察



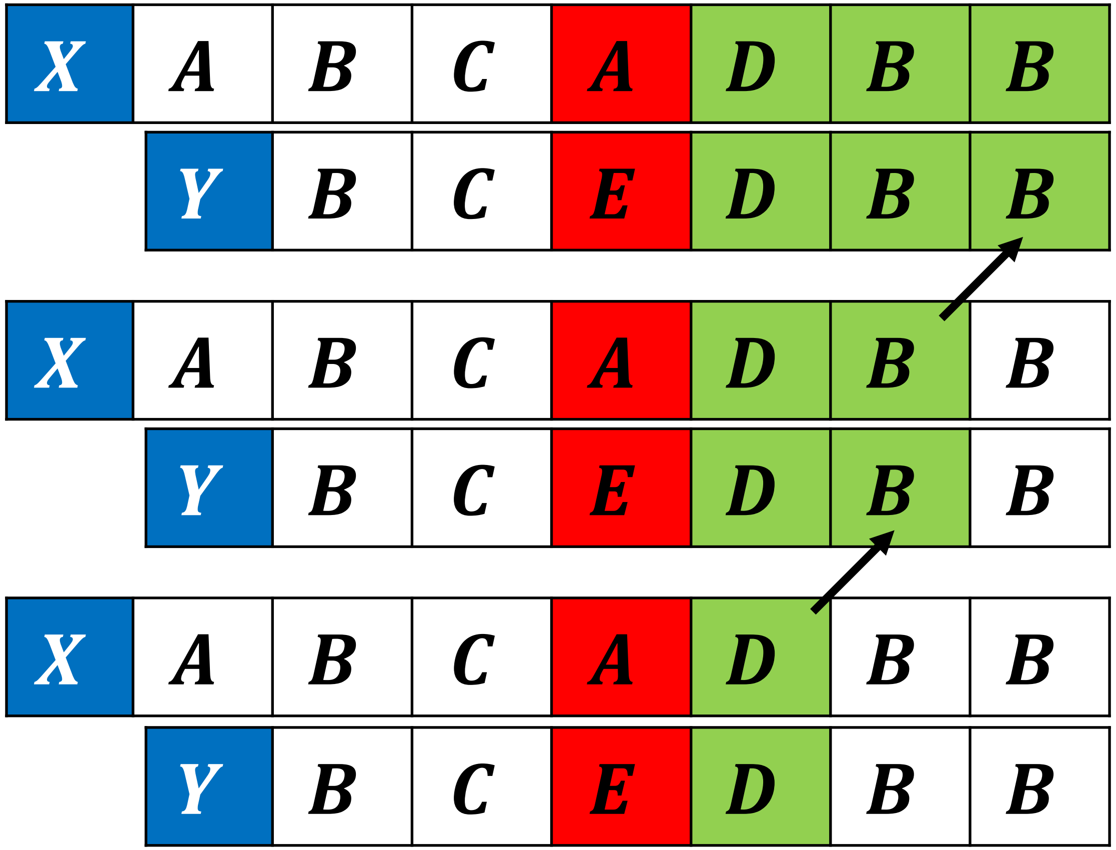
枚举观察

X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

X	*A*	*B*	*C*	*A*	*D*	*B*	*B*
	Y	*B*	*C*	*E*	*D*	*B*	*B*

枚举观察



- 可能存在**最优子结构**和**重叠子问题**

问题：如何利用动态规划求解？

问题结构分析

- 给出问题表示
  - $C[i,j]$
    - $X[1..i]$和$Y[1..j]$中，以x_i和y_j结尾的最长公共子串$Z[1..l]$的长度

X_i	x_1	x_2	…	x_{i-1}	x_i
Y_j	y_1	y_2	…	y_{j-1}	y_j

Z_l	z_1	…	z_{l-1}	z_l

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

问题结构分析

- 给出问题表示
  - $C[i,j]$
    - $X[1..i]$和$Y[1..j]$中，以x_i和y_j结尾的最长公共子串$Z[1..l]$的长度

X_i	x_1	x_2	…	x_{i-1}	x_i
Y_j	y_1	y_2	…	y_{j-1}	y_j

x_i

$\parallel$

Z_l	z_1	…	z_{l-1}	z_l

$\parallel$

y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

问题结构分析

- 给出问题表示
  - $C[i,j]$
    - $X[1..i]$和$Y[1..j]$中，以x_i和y_j结尾的最长公共子串$Z[1..l]$的长度

X_i	x_1	x_2	…	x_{i-1}	x_i
Y_j	y_1	y_2	…	y_{j-1}	y_j

x_i

$\|$

Z_l	z_1	…	z_{l-1}	z_l

$\|$

y_j

- 明确原始问题
  - $p_{max} = \max\limits_{1\le i\le n, 1\le j\le m}\{C[i,j]\}$
    - $X[1..n]$和$Y[1..m]$中最长公共子串的长度

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_7 \neq y_6$

X	A	B	C	A	D	B	B

Y	B	C	E	D	B	A

- 情况2：$x_7 = y_6$

X	A	B	C	A	D	B	B

Y	B	C	E	D	B

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	A	D	B	B
	Y	B	C	E	D	B	A

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_7 \neq y_6$

$C[7,6]$

X	A	B	C	A	D	B	B
	Y	B	C	E	D	B	A

不存在以其结尾的公共子串

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_7 \neq y_6$

$$C[7,6] = 0$$

X	A	B	C	A	D	B	B
	Y	B	C	E	D	B	A

不存在以其结尾的公共子串

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_i \neq y_j$

$$C[i,j] = 0$$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况1：$x_i \neq y_j$

$$C[i,j] = 0$$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

无子问题

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	A	D	B	B
	Y	B	C	E	D	B	B

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况2：$x_7 = y_6$

$C[7,6]$

X	A	B	C	A	D	B	B
	Y	B	C	E	D	B	B

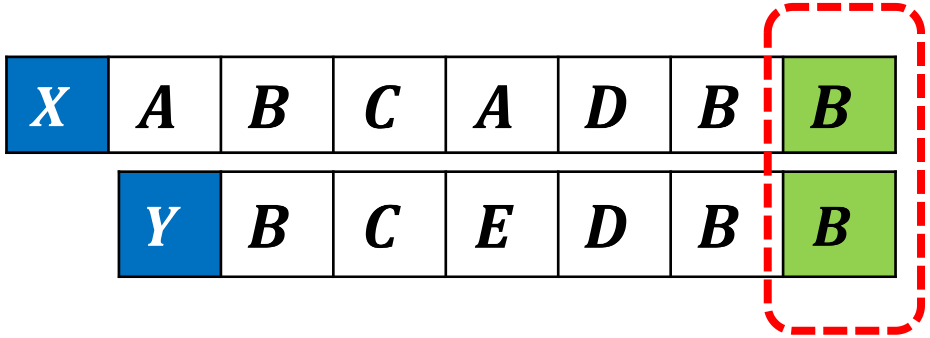
存在以其结尾的公共子串

- 问题结构分析
- **递推关系建立**
- 自底向上计算
- 最优方案追踪

递推关系建立：分析最优（子）结构

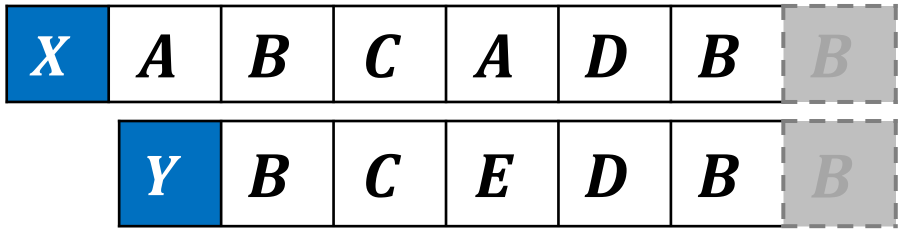
- 情况2：$x_7 = y_6$

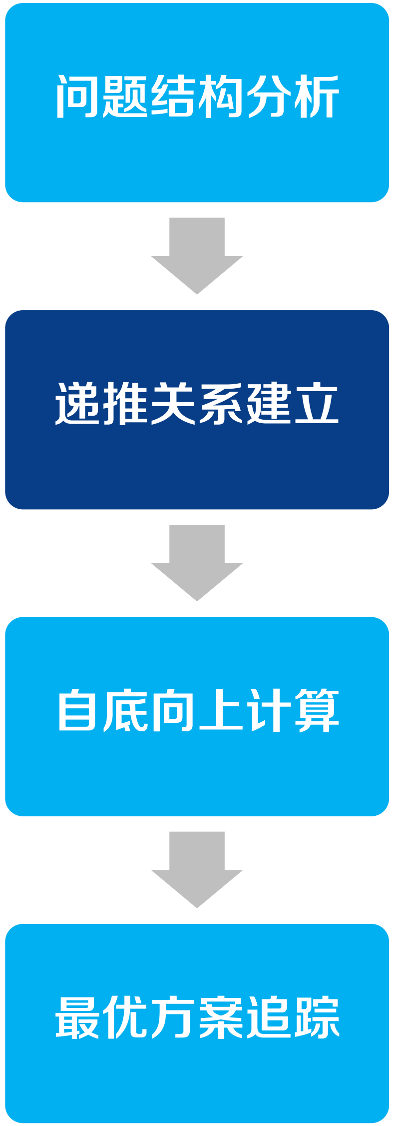
$C[7,6]$



存在以其结尾的公共子串

$C[7,6] = C[7-1, 6-1] + 1$





递推关系建立：分析最优（子）结构

- 情况2：$x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况2：$x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$\longrightarrow$

$C[i,j] = C[i-1,j-1] + 1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

递推关系建立：分析最优（子）结构

- 情况2：$x_i = y_j$

$C[i,j]$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

$C[i,j] = C[i-1,j-1] + 1$

X	x_1	x_2	…	x_{i-1}	x_i
Y	y_1	y_2	…	y_{j-1}	y_j

- $\boxed{C[i,j]} = \boxed{C[i-1,j-1]} + 1$

最优子结构

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

递推关系建立：构造递推公式

- $C[i,j]=\begin{cases}0 & ,x_i\neq y_j\\ C[i-1,j-1]+1 & ,x_i=y_j\end{cases}$

$$\begin{array}{c}\boxed{C[i,j]}\\ \Big\uparrow{\scriptstyle +1}\\ \boxed{C[i-1,j-1]}\end{array}$$

问题结构分析

↓

递推关系建立

↓

自底向上计算

↓

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0]=C[0,j]=0$
    - 某序列长度为0时，最长公共子串为0

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子串为0

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0				
$i=n$	0				

初始化

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：确定计算顺序

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子串为0
- 递推公式
  - $C[i,j] = \begin{cases} 0 & ,x_i \neq y_j \\ C[i-1,j-1]+1 & ,x_i = y_j \end{cases}$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0				
…	0			$C[i,j]$	
$i=n$	0				

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：依次求解问题

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子串为0
- 递推公式
  - $$C[i,j] = \begin{cases} 0 & ,x_i \neq y_j \\ C[i-1,j-1]+1 & ,x_i = y_j \end{cases}$$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	
$i=1$	0				
$i=2$	0				
…	0				
$i=n$	0				

自底向上计算

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

自底向上计算：依次求解问题

- 初始化
  - $C[i,0] = C[0,j] = 0$
    - 某序列长度为0时，最长公共子串为0
- 原始问题
  - $p_{max} = \max_{1\le i\le n, 1\le j\le m}\{C[i,j]\}$

$C[i,j]$	$j=0$	$j=1$	$j=2$	…	$j=m$
$i=0$	0	0	0	0	0
$i=1$	0				
$i=2$	0		★		
…	0		最优解		
$i=n$	0				

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

最优方案追踪

- 记录决策过程
  - 最长公共子串末尾位置为p_{max}
  - 最长公共子串长度为l_{max}

问题结构分析
↓
递推关系建立
↓
自底向上计算
↓
最优方案追踪

最优方案追踪

- 记录决策过程
  - 最长公共子串末尾位置为p_{max}
  - 最长公共子串长度为l_{max}
- 输出最优方案
  - 最长公共子串$< x_{p_{max}-l+1}, x_{p_{max}-l+2}, \dots, x_{p_{max}} >$

X	x_1	$\dots$	$x_{p_{max}-l+1}$	$\dots$	$x_{p_{max}}$	$\dots$	x_n

长度为l_{max}

问题结构分析

递推关系建立

自底向上计算

最优方案追踪

算法实例

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

算法实例

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

位置$p_{max} = 0$

长度$l_{max} = 0$

$C[\,]$

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

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

i \ 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						

初始化

算法实例

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

$x_i \neq y_j$

位置$p_{max} = 0$

长度$l_{max} = 0$

$C[\ ]$

$i \backslash j$	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						

算法实例

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

$x_i \neq y_j$

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

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

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

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

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

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

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

i \ 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						

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\ ]$

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

算法实例

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

位置$p_{max} = 0$
长度$l_{max} = 0$

$C[\,]$

$i \backslash j$	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						

算法实例

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

$x_i = y_j$

位置$p_{max} = 0$

长度$l_{max} = 0$

$C[\,]$

j \ i	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						

算法实例

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

$x_i = y_j$

位置p_{max} = 2

长度l_{max} = 1

$C[\ ]$

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

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\ ]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\,]$

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	
3	0						
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\,]$

$i \backslash j$	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						
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 2$
长度$l_{max} = 1$

$C[\,]$

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					
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max}=3$
长度$l_{max}=2$

$C[\ ]$

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				
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$

长度$l_{max} = 2$

$C[\,]$

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			
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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		
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\ ]$

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	
4	0						
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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					
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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				
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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			
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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		
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\ ]$

j / i	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	
5	0						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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					
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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				
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max}=3$
长度$l_{max}=2$

$C[\,]$

$i \backslash j$	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	
6	0						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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						
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\ ]$

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					
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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				
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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			
7	0						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\ ]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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						

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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					

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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	0				

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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			

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

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		

算法实例

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

位置$p_{max} = 3$
长度$l_{max} = 2$

$C[\,]$

$i \backslash j$	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	0	0	0	1	

算法实例

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

位置p_{max} = 7

长度l_{max} = 3

$C[\,]$

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	1	3

算法实例

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

位置$p_{max} = 7$
长度$l_{max} = 3$

$C[\,]$

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	[illegible]	[illegible]	[illegible]	1
7	0	1	0	0	0	1	3

最长公共子串长度

算法实例

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

位置$p_{max} = 7$
长度$l_{max} = 3$

$C[\,]$

j \ i	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	1	3

伪代码

- **Longest-Common-Substring(X, Y)**

输入: 两个字符串X, Y
输出: X和Y的最长公共子串
//初始化
$n \leftarrow \text{length}(X)$
$m \leftarrow \text{length}(Y)$
新建二维数组$C[0..n, 0..m]$
$l_{max} \leftarrow 0$
$p_{max} \leftarrow 0$
for $i \leftarrow 0$ *to* n **do**
| $C[i, 0] \leftarrow 0$
end
for $j \leftarrow 0$ *to* m **do**
| $C[0, j] \leftarrow 0$
end

序列长度

伪代码

- **Longest-Common-Substring(X, Y)**

输入: 两个字符串X, Y
输出: X和Y的最长公共子串
//初始化
$n \leftarrow$length(X)
$m \leftarrow$length(Y)
新建二维数组$C[0..n, 0..m]$
$l_{max} \leftarrow 0$
$p_{max} \leftarrow 0$
for $i \leftarrow 0$ *to* n **do**
| $C[i, 0] \leftarrow 0$
end
for $j \leftarrow 0$ *to* m **do**
| $C[0, j] \leftarrow 0$
end

初始化最优解

伪代码

- **Longest-Common-Substring(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i ≠ Y_j then
            C[i, j] ← 0
        end
        else
            C[i, j] ← C[i − 1, j − 1] + 1
            if C[i, j] > l_max then
                l_max ← C[i, j]
                p_max ← i
            end
        end
    end
end
return l_max, p_max
```

依次计算子问题

伪代码

- **Longest-Common-Substring(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i ≠ Y_j then
            C[i, j] ← 0
        end
        else
            C[i, j] ← C[i − 1, j − 1] + 1
            if C[i, j] > l_max then
                l_max ← C[i, j]
                p_max ← i
            end
        end
    end
end
return l_max, p_max
```

末尾不等

伪代码

- **Longest-Common-Substring(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i ≠ Y_j then
            C[i, j] ← 0
        end
        else
            C[i, j] ← C[i − 1, j − 1] + 1
            if C[i, j] > l_max then
                l_max ← C[i, j]
                p_max ← i
            end
        end
    end
end
return l_max, p_max
```

末尾相等

伪代码

- **Longest-Common-Substring(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i ≠ Y_j then
            C[i, j] ← 0
        end
        else
            C[i, j] ← C[i − 1, j − 1] + 1
            if C[i, j] > l_max then
                l_max ← C[i, j]
                p_max ← i
            end
        end
    end
end
return l_max, p_max
```

记录最长公共子串

伪代码

- **Print-LCS(X, l_{max}, p_{max})**

输入: **字符串** X, l_{max}, p_{max}
输出: X **和** Y**的最长公共子串**
if $l_{max} = 0$ **then**
  return $NULL$
end
for $i \leftarrow (p_{max} - l_{max} + 1)$ *to* p_{max} **do**
  print X_i
end

无公共子串

伪代码

- **Print-LCS(X, l_{max}, p_{max})**

输入: 字符串 X, l_{max}, p_{max}
输出: X 和 Y的最长公共子串

if $l_{max} = 0$ **then**
  return $NULL$
end
for $i \leftarrow (p_{max} - l_{max} + 1)$ *to* p_{max} **do**
  print X_i
end

追踪最优解

时间复杂度分析

- **Longest-Common-Substring(X, Y)**

```
//动态规划
for i ← 1 to n do
    for j ← 1 to m do
        if X_i ≠ Y_j then
            C[i, j] ← 0
        end
        else
            C[i, j] ← C[i − 1, j − 1] + 1
            if C[i, j] > l_max then
                l_max ← C[i, j]
                p_max ← i
            end
        end
    end
end
return l_max, p_max
```

时间复杂度：$O(n \cdot m)$

小结

最长公共子序列

X	D	B	D	A	B
Y	C	A	B	B	

最长公共子串

X	D	B	D	A	B
Y	C	A	B	B	

小结

最长公共子序列

X	D	B	D	A	B
Y	C	A	B	B	

最长公共子串

X	D	B	D	A	B
Y	C	A	B	B	

情况2：$x_5 = y_4$

小结

最长公共子序列

X	D	B	D	A	B
Y	C	A	B	B	

最长公共子串

X	D	B	D	A	B
Y	C	A	B	B	

小结

最长公共子序列

X	D	B	D	A	B
Y	C	A	B	B	

X	D	B	D	A	B
Y	C	A	B	B	

最长公共子串

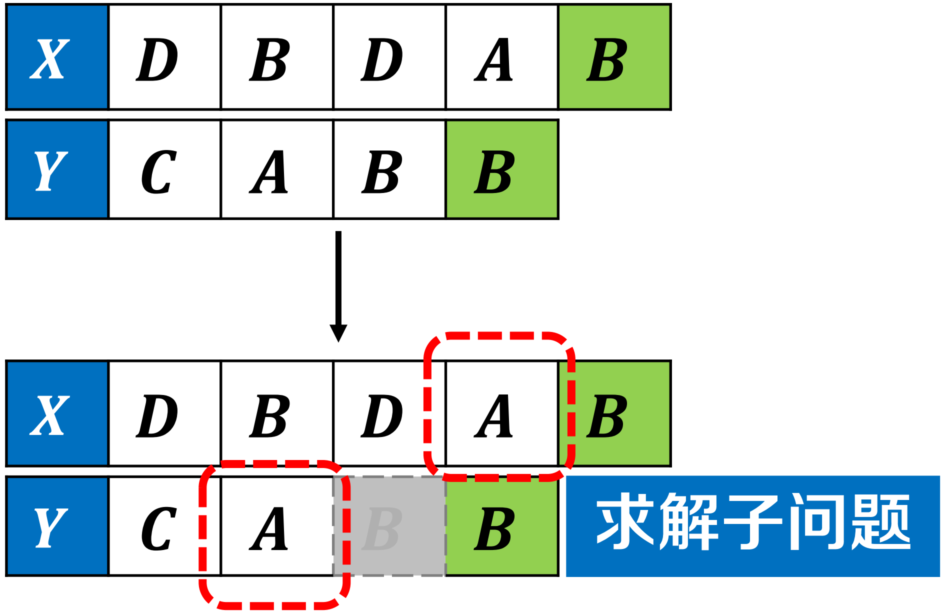
X	D	B	D	A	B
Y	C	A	B	B	

X	D	B	D	A	B
Y	C	A	B	B	

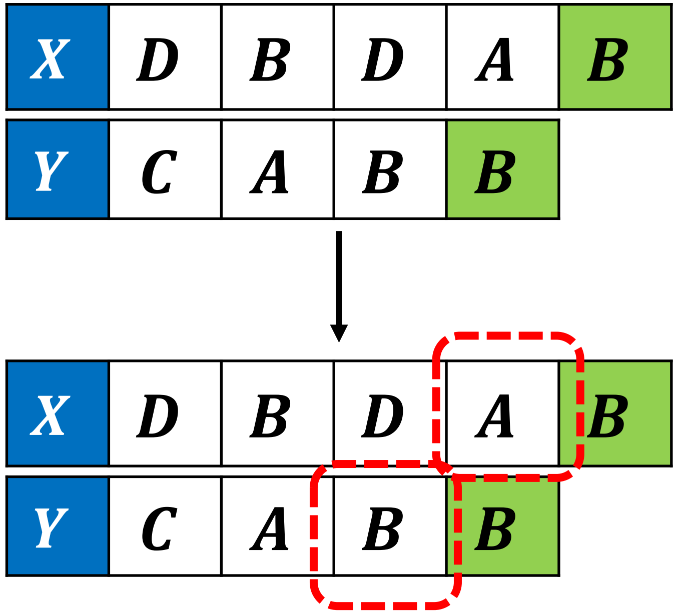
情况1： $x_4 \neq y_3$

小结

最长公共子序列

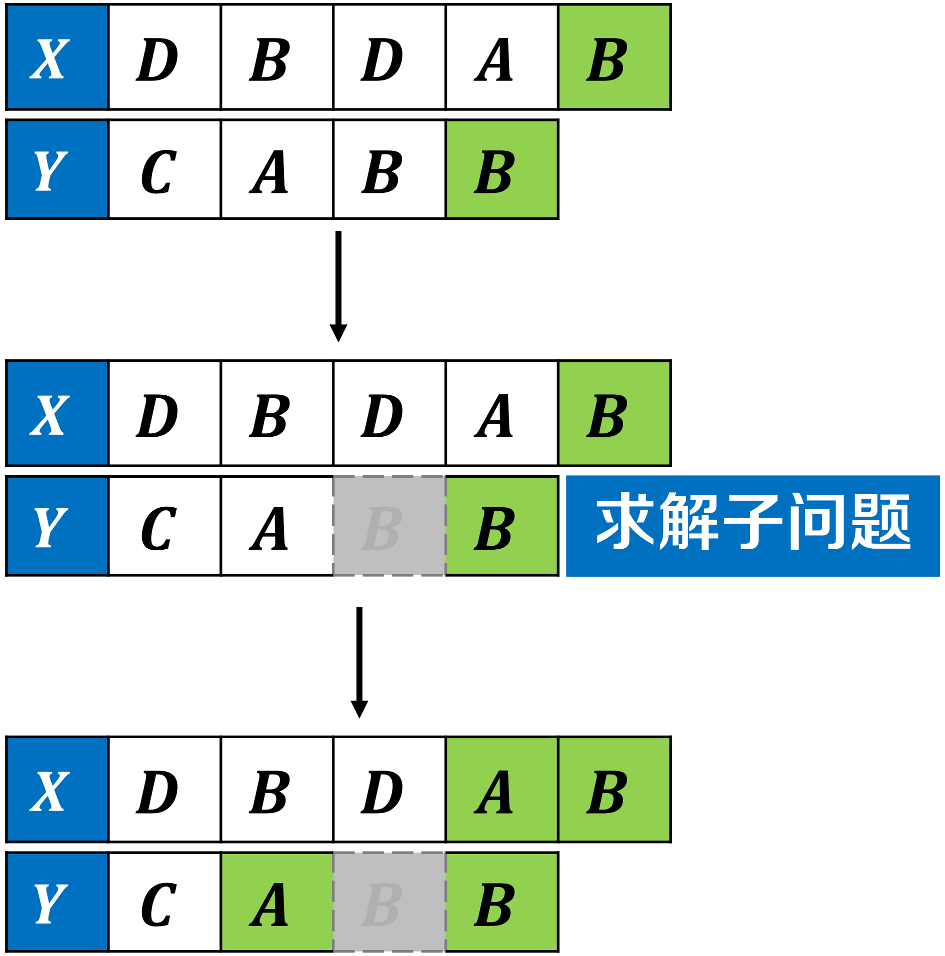


最长公共子串

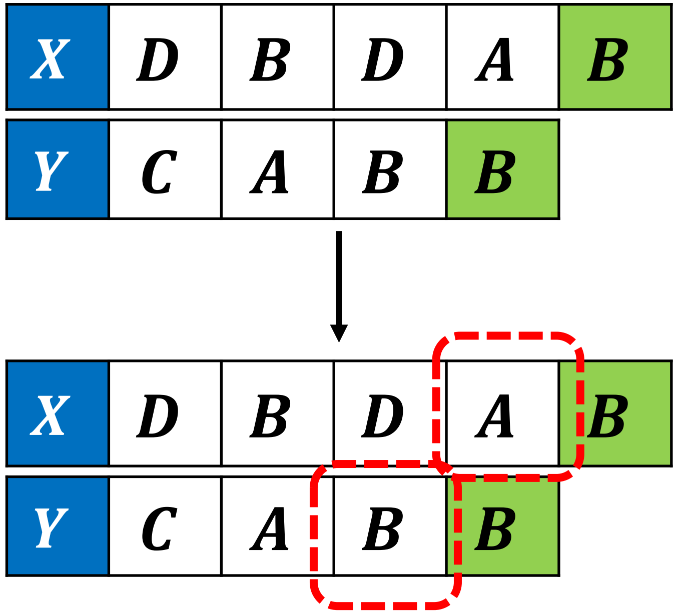


小结

最长公共子序列

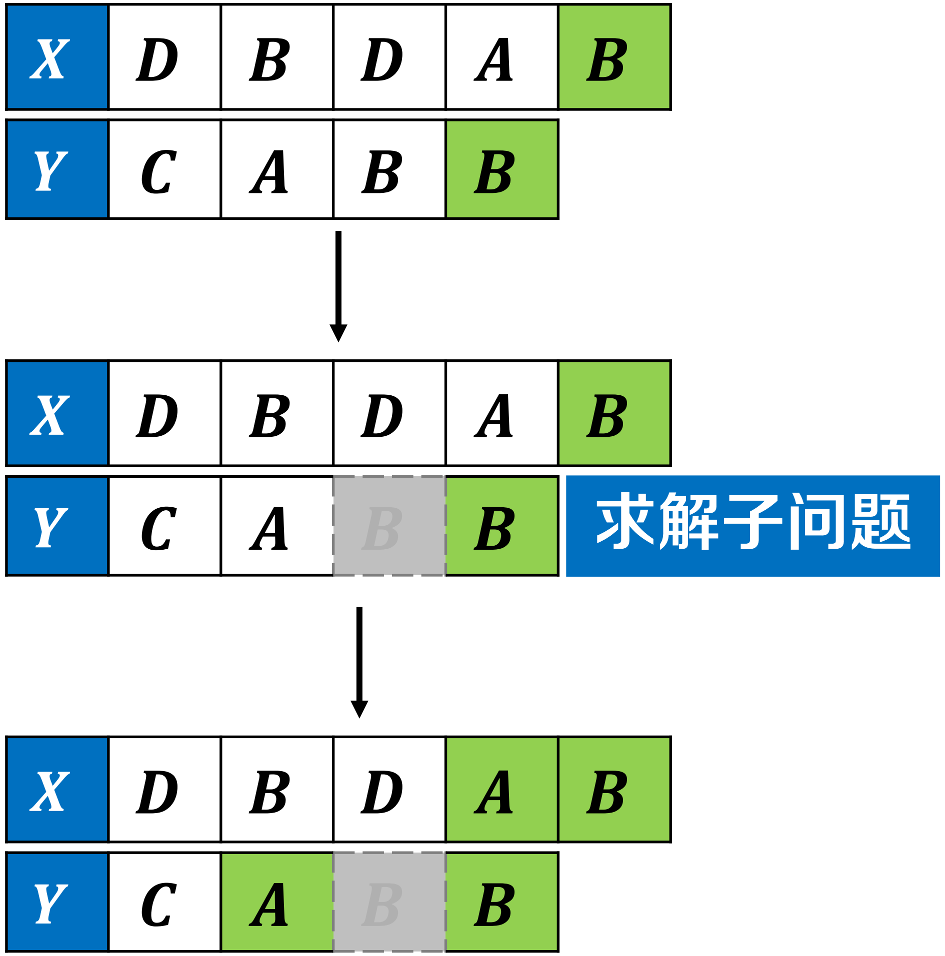


最长公共子串



小结

最长公共子序列



最长公共子串