

Fundamentos de análisis y diseño de algoritmos

Programación dinámica

LCS (Longest Common Subsequence)

Multiplicación de matrices

Propiedades generales de la programación dinámica

Programación dinámica

Dada una secuencia $X = \langle x_1, x_2, \dots, x_m \rangle$, se dice que $Z = \langle z_1, z_2, \dots, z_n \rangle$ es una **subsecuencia de X** si los símbolos z_i se dan en X siguiendo el orden que se dan en Z (no necesariamente uno seguido de otro)

Por ejemplo, para $X = \langle A, B, C, B, D, A, B \rangle$, algunas subsecuencias son:

$\langle A, B, C \rangle$,

$\langle A, B, C, D \rangle$

$\langle B, C, D, B \rangle$

Programación dinámica

Por ejemplo, para $X = \langle A, \overset{\cdot}{B}, C, B, D, \overset{\cdot}{A}, B \rangle$, algunas subsecuencias son:

$\langle A, B, C \rangle,$

$\langle A, B, C, D \rangle$

$\langle B, C, D, B \rangle$

Sin embargo, no son subsecuencias:

$\langle A, B, \overset{\circ}{A}, A \rangle, \quad \times$

$\langle B, A, D \rangle \quad \times$

$\langle D, A, D \rangle \quad \times$

Programación dinámica

Problema; Dadas dos secuencias $X=\langle x_1, x_2, \dots, x_m \rangle$ y $Y=\langle y_1, y_2, \dots, y_n \rangle$, encontrar la(s) subsecuencia(s) más larga(s) entre a X y Y

$X=ABCBDAB$

$Y=BDCABA$

Indique algunas subsecuencias

Indique una solución al problema

Programación dinámica

Problema; Dadas dos secuencias $X=\langle x_1, x_2, \dots, x_m \rangle$ y $Y=\langle y_1, y_2, \dots, y_n \rangle$, encontrar la(s) subsecuencia(s) más larga(s) entre a X y Y

X=ABCBDAB

Y=BDCABA

Algunas subsecuencias son: A, CB, ABA, BDAB, CAB, DAB

El problema de la subsecuencia más larga tiene varias soluciones: BCBA, BCAB y BDAB

Programación dinámica

Problema; Dadas dos secuencias $X = \langle x_1, x_2, \dots, x_m \rangle$ y $Y = \langle y_1, y_2, \dots, y_n \rangle$, encontrar la(s) subsecuencia(s) más larga(s) entre a X y Y

¿Cuál podría ser una solución al problema?

Programación dinámica

Problema; Dadas dos secuencias $X = \langle x_1, x_2, \dots, x_m \rangle$ y $Y = \langle y_1, y_2, \dots, y_n \rangle$, encontrar la(s) subsecuencia(s) más larga(s) entre a X y Y

$$\{a, b, c\} = \{\emptyset, \{a\}, \{b\}, \{c\}, \\ \{a, b\}, \{b, c\}, \{a, c\}, \\ \{a, b, c\}\} \quad O(2^n)$$

¿Cuál podría ser una solución al problema?

Enumerar todas las subsecuencias de X y examinar si es también una subsecuencia de Y. Tomar al final la subsecuencia más larga

Existen 2^m posibles suconjuntos (subsecuencias) del conjunto de m elementos \rightarrow Tiempo exponencial

Programación dinámica

Problema; Dadas dos secuencias $X = \langle x_1, x_2, \dots, x_m \rangle$ y $Y = \langle y_1, y_2, \dots, y_n \rangle$, encontrar la(s) subsecuencia(s) más larga(s) entre a X y Y

La solución al problema presenta una característica especial

$X = A B C B D A B$

$Y = B D C A B A$

$n - 1$

$$T(n) = 2T(n-1) + 1$$

$$n=2 \quad O(2^n)$$

$B D A B$

$X_{1..7} \quad Y_{1..6}$

$B D A \quad X = A B C B D A \quad Y = B D C A$
 $1_{1..6} \quad Y_{1..4}$

$B D \quad X = A B C B D \quad Y = B D C$
 $X_{1..5} \quad Y_{1..3}$

$B \quad X = A B C B \quad Y = B$
 $X_{1..4} \quad Y_{1..1}$

Programación dinámica

Problema; Dadas dos secuencias $X = \langle x_1, x_2, \dots, x_m \rangle$ y $Y = \langle y_1, y_2, \dots, y_n \rangle$, encontrar la *subsecuencia más larga (LCS)* entre X y Y

- Si $x_m = y_n$, la solución del problema $LCS(X_m, Y_n)$ será $LCS(X_{m-1}, Y_{n-1})$ y pegar al final x_m
- Si $x_m \neq y_n$, la solución del problema $LCS(X_m, Y_n)$ se escoge entre $LCS(X_{m-1}, Y_n)$ y $LCS(X, Y_{n-1})$, esto es, de estas dos se elige aquella de mayor longitud

Programación dinámica

Si $x_m = y_n$, la solución del problema $LCS(X_m, Y_n)$ será $LCS(X_{m-1}, Y_{n-1})$ y pegar al final x_m

- Si $x_m \neq y_n$, la solución del problema $LCS(X_m, Y_n)$ se escoge entre $LCS(X_{m-1}, Y_n)$ y $LCS(X, Y_{n-1})$, esto es, de estas dos se elige aquella de mayor longitud

$LCS(ABD, ACD)$. . .

$$LCS(AB\boxed{D}, A\boxed{C}\boxed{D})$$



$$LCS(AB, \overset{A}{\cancel{AC}}) + D$$

AD

$$\max(LCS(A, \overset{A}{\cancel{AC}}))$$

$$LCS(AB, \overset{A}{\cancel{A}})$$

$$\max(\overset{+0}{\cancel{LCS(\boxed{D}, AC)}}, \overset{+1}{\cancel{LCS(A, A)}})$$

$$\max(\overset{+1}{\cancel{LCS(A, A)}}, \overset{+0}{\cancel{LCS(AB, \boxed{D})}})$$

Programación dinámica

Si $x_m = y_n$, la solución del problema $LCS(X_m, Y_n)$ será $LCS(X_{m-1}, Y_{n-1})$ y pegar al final x_m

- Si $x_m \neq y_n$, la solución del problema $LCS(X_m, Y_n)$ se escoge entre $LCS(X_{m-1}, Y_n)$ y $LCS(X, Y_{n-1})$, esto es, de estas dos se elige aquella de mayor longitud

$LCS(ABD, ACD)$

solucion= $LCS(AB, AC) + D$

Programación dinámica

Si $x_m = y_n$, la solución del problema $LCS(X_m, Y_n)$ será $LCS(X_{m-1}, Y_{n-1})$ y pegar al final x_m

- Si $x_m \neq y_n$, la solución del problema $LCS(X_m, Y_n)$ se escoge entre $LCS(X_{m-1}, Y_n)$ y $LCS(X, Y_{n-1})$, esto es, de estas dos se elige aquella de mayor longitud

$LCS(ABD, ACD)$

solucion= $LCS(AB, AC) + D$

solucion'= $LCS(A, AC)$ solucion'= $LCS(AB, A)$

Programación dinámica

• $LCS(ABD, ACD)$

solucion = $LCS(AB, AC) + D$

solucion' = $LCS(A, AC)$ solucion' = $LCS(AB, A)$

solucion'' = $LCS(, AC)$ solucion'' = $LCS(A, A)$

Programación dinámica

• $LCS(ABD, ACD)$

solucion = $LCS(AB, AC) + D$

solucion' = $LCS(A, AC)$ solucion' = $LCS(AB, A)$

solucion'' = $LCS(, AC)$ solucion'' = $LCS(A, A)$

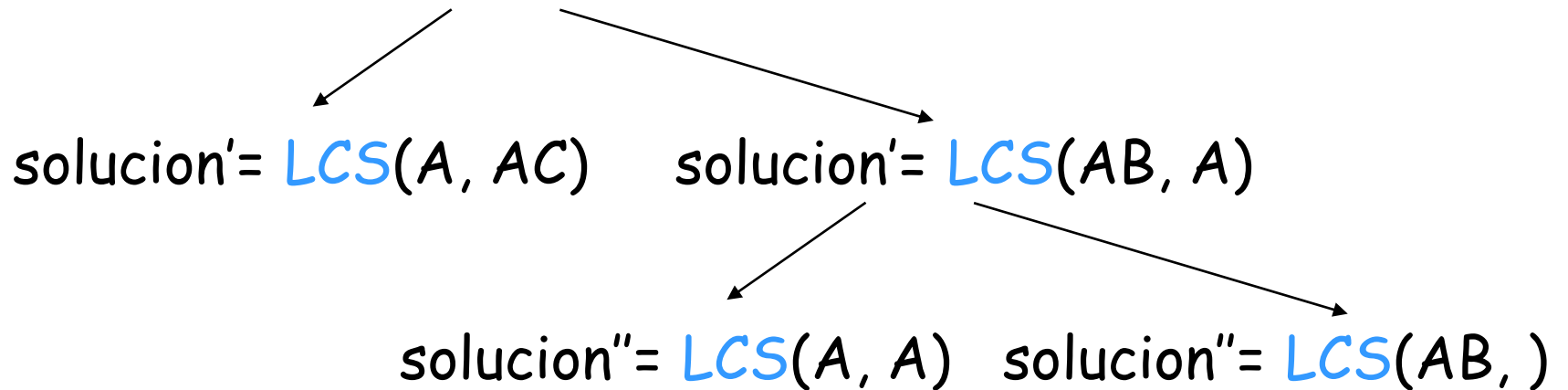
$LCS(A, A) = LCS(,) + A$

Se obtiene AD como una solución

Programación dinámica

• $LCS(ABD, ACD)$

solucion = $LCS(AB, AC) + D$



Ya fue calculado, no se calcula nuevamente.

Se retorna A

Se obtiene AD como una solución

Programación dinámica

Note que hay subproblemas que se sobrelapan, para encontrar $LCS(X_m, Y_n)$ se podría necesitar encontrar $LCS(X_{m-1}, Y_n)$ y $LCS(X_m, Y_{n-1})$, pero cada uno de los subproblemas tendrá que encontrar $LCS(X_{m-1}, Y_{n-1})$

Cuando se comparten subproblemas, no se calculan nuevamente, ya se conoce el valor, simplemente se utiliza

Estrategia: Divide y vencerás donde los subproblemas no son independientes sino que algunos son compartidos. Esto se aprovecha para no tener que hacer cálculos repetidos

Programación dinámica

Se tiene una función de costo que permite conocer la longitud de la solución de óptima:

1) Subproblemas

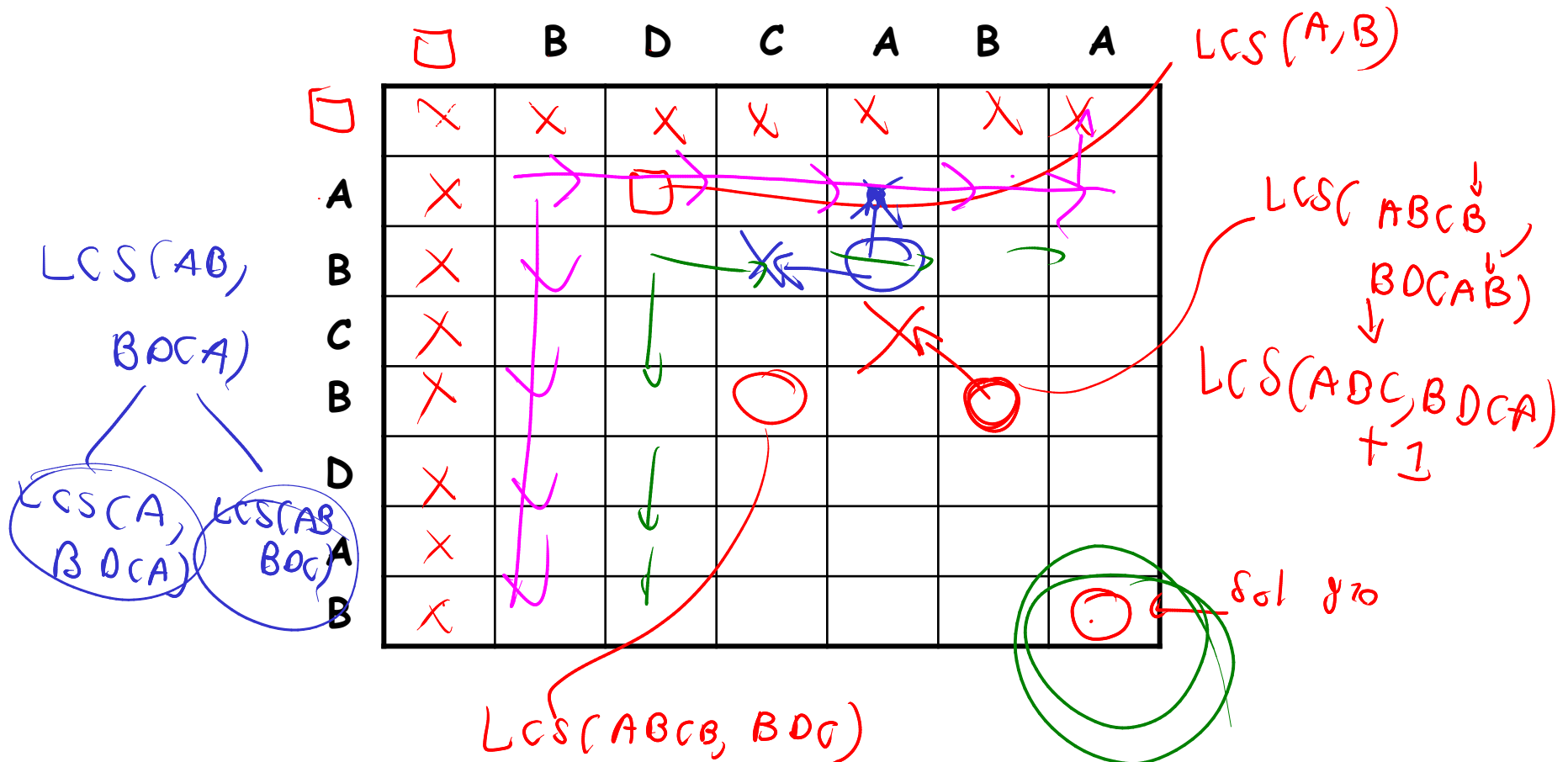
2) Descripción

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i = x_j \text{ e } i,j > 0 \\ \max(c[i,j-1], c[i-1,j]) & , \text{ si } x_i \neq x_j \text{ e } i,j > 0 \end{cases}$$

c es una matriz que indica la longitud de la secuencia más larga entre X_i y Y_j

Programación dinámica

LCS(ABCBDAB, BDCABA)



Programación dinámica

LCS(ABCBBDAB, BDCABA)

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0								
1	A							
2	B							
3	C							
4	B							
5	D							
6	A							
7	B							

Programación dinámica

LCS(ABCBDAB, BDCABA)

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0			?					
1	A							
2	B							
3	C							
4	B							
5	D							
6	A							
7	B							

$C[0,1]$ es la longitud de la subsecuencia más larga entre las secuencias $X=""$ y $Y="B"$

Programación dinámica

LCS(ABCBDBAB, BDCABA)

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0			0	?				
1	A							
2	B							
3	C							
4	B							
5	D							
6	A							
7	B							

$C[0,1]$ es la longitud de la subsecuencia más larga entre las secuencias $X=""$ y $Y="B"$

Programación dinámica

LCS(ABCBDAB, BDCABA)

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0			0	0	0	0	0	0
1	A	0						
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	?					
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	?					
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i = x_j \text{ e } i,j > 0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j > 0 \end{cases}$$

Programación dinámica

	0	1	2	3	4	5	6
		B	D	C	A	B	A
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

BCBA

BCAB

BDAB

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	?	?			
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	?		
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	?	
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

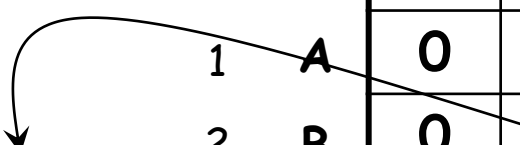
		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	?					
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1					
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

Longitud para LCS(AB,B)



$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	?				
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6	
			B	D	C	A	B	A	
0		0	0	0	0	0	0	0	
1	A	0	0	0	0	1	1	1	Longitud para LCS(A,BD)
2	B	0	1	1					
3	C	0							
4	B	0							Longitud para LCS(AB,BD)
5	D	0							
6	A	0							
7	B	0							

Longitud para LCS(AB,B)

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i = x_j \text{ e } i,j > 0 \\ \max(c[i,j-1], c[i-1,j]) & , \text{ si } x_i \neq x_j \text{ e } i,j > 0 \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1				
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

Completar la tabla

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Qué significado tiene el hecho de que $c[4,4]=2$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Qué significado tiene el hecho de que $c[4,4]=2$

Indica que la longitud de $LCS(ABCB, BDCA)$ es 2

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Qué significado tiene el hecho de que $c[7,4]=3$

Indica que la longitud de $LCS(ABCB DAB, BDCA)$ es 3

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Qué casilla de la matriz guarda la longitud de la solución al problema original

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Cuál es la solución, es decir, cual es la subsecuencia común?

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0↑	0↑	0↑	1↖	1←	1↖
2	B	0	1↖	1←	1←	1↑	2↖	2←
3	C	0	1↑	1↑	2↖	2←	2↑	2↑
4	B	0	1↖	1↑	2↑	2↑	3↖	3←
5	D	0	1↑	2↖	2↑	2↑	3↑	3↑
6	A	0	1	2↑	2↑	3↖	3↑	4↖
7	B	0	1↖	2↑	2↑	3↑	4↖	4↑

Las direcciones se guardan en otro arreglo llamado B

$$c[i,j] = \begin{cases} 0 & , \text{ si } i=0 \text{ o } j=0 \\ c[i-1,j-1] + 1 & , \text{ si } x_i=x_j \text{ e } i,j>0 \text{ (↖)} \\ \max(c[i,j-1], c[i-1,j]), & \text{ si } x_i \neq x_j \text{ e } i,j>0 \text{ (←) (↑)} \end{cases}$$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0↑	0↑	0↑	←1	←1	←1
2	B	0	←1	←1	←1	1↑	←2	←2
3	C	0	1↑	1↑	←2	←2	2↑	2↑
4	B	0	←1	1↑	2↑	2↑	←3	←3
5	D	0	1↑	←2	2↑	2↑	3↑	3↑
6	A	0	1	2↑	2↑	←3	3↑	←4
7	B	0	←1	2↑	2↑	3↑	←4	4↑

Solo ↖ indica que es un símbolo común, éstos se imprimen.
 En los demás casos, se sigue la flecha, ↑ o ←

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0↑	0↑	0↑	1↖	1←	1↖
2	B	0	1↖	1←	1←	1↑	2↖	2←
3	C	0	1↑	1↑	2↖	2←	2↑	2↑
4	B	0	1↖	1↑	2↑	2↑	3↖	3←
5	D	0	1↑	2↖	2↑	2↑	3↑	3↑
6	A	0	1	2↑	2↑	3↖	3↑	4↖
7	B	0	1↖	2↑	2↑	3↑	4↖	4↑

Muestre la solución al problema

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0↑	0↑	0↑	1↖	1←	1↖
2	B	0	1↖	1↖	1←	1↑	2↖	2←
3	C	0	1↑	1↑	2↖	2↖	2↑	2↑
4	B	0	1↖	1↑	2↑	2↑	3↖	3←
5	D	0	1↑	2↖	2↑	2↑	3↑	3↑
6	A	0	1	2↑	2↑	3↖	3↑	4↖
7	B	0	1↖	2↑	2↑	3↑	4↖	4↑

ABCB, se invierte y se obtiene BCBA

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0↑	0↑	0↑	1↖	1←	1↖
2	B	0	1↖	1←	1←	1↑	2↖	2←
3	C	0	1↑	1↑	2↖	2←	2↑	2↑
4	B	0	1↖	1↑	2↑	2↑	3↖	3
5	D	0	1↑	2↖	2↑	2↑	3↑	3↑
6	A	0	1	2↑	2↑	3↖	3↑	4↖
7	B	0	1↖	2↑	2↑	3↑	4↖	4↑

Muestre la solución al problema $LCS(ABCB, BDCABA)$

Programación dinámica

		0	1	2	3	4	5	6
			B	D	C	A	B	A
0		0	0	0	0	0	0	0
1	A	0	0	0	0	1	1	1
2	B	0	1	1	1	1	2	2
3	C	0	1	1	2	2	2	2
4	B	0	1	1	2	2	3	3
5	D	0	1	2	2	2	3	3
6	A	0	1	2	2	3	3	4
7	B	0	1	2	2	3	4	4

Muestre la solución al problema $LCS(ABCB, BDCABA)$
 BCB, que en orden invertido es BCB

LCS-LENGTH(X,Y)

$m \leftarrow \text{length}[X]$

$n \leftarrow \text{length}[Y]$

for $i \leftarrow 1$ to m

do $c[i,0] \leftarrow 0$

for $j \leftarrow 0$ to n

do $c[0,j] \leftarrow 0$

for $i \leftarrow 1$ to m

do for $j \leftarrow 1$ to n

do if $x_i = y_j$

then $c[i,j] \leftarrow c[i-1,j-1] + 1$

$b[i,j] \leftarrow "$ ↖ $"$

else if $c[i-1,j] \geq c[i,j-1]$

then $c[i,j] \leftarrow c[i-1,j]$

$b[i,j] \leftarrow "$ ↑ $"$

else $c[i,j] \leftarrow c[i,j-1]$

$b[i,j] \leftarrow "$ ← $"$

return c and b

Indique la longitud de
 $\text{LCS}(\text{AFCEA}, \text{CFEHA})$

PRINT-LCS(b, X, i, j)

if $i=0$ or $j=0$

then return

if $b[i,j]=$ " ↖ "

then PRINT-LCS(b, X, $i-1$, $j-1$)

print x_i

else if $b[i,j]=$ " ↑ "

then PRINT-LCS(b, X, $i-1$, j)

else PRINT-LCS(b, X, i , $j-1$)

Resuelva LCS(AFCEA, CFEHA)