

Nejdelší společná podposloupnost

Dvě
posloupnosti

A: C B E A D D E A $|A| = 8$

B: D E C D B D A $|B| = 7$

Společná
podposloupnost

A: C B E A D D E A

B: D E C D B D A

C: C D A $|C| = 3$

Nejdelší
společná
podposloupnost
(NSP)

A: C B E A D D E A

B: D E C D B D A

C: E D D A $|C| = 4$

Nejdelší společná podposloupnost

$A_n: (a_1, a_2, \dots, a_n)$

$B_m: (b_1, b_2, \dots, b_m)$

$C_k: (c_1, c_2, \dots, c_k)$

.....
 $C_k = \text{LCS}(A_n, B_m)$

	1	2	3	4	5	6	7	8
$A_8:$	C	B	E	A	D	D	E	A
$B_7:$	D	E	C	D	B	D	A	
$C_4:$	E	D	D	A				

Rekurzivní pravidla:

$(a_n = b_m) \implies (c_k = a_n = b_m) \ \& \ (C_{k-1} = \text{LCS}(A_{n-1}, B_{m-1}))$

	1	2	3	4	5	6	7	8
$A_8:$	C	B	E	A	D	D	E	A
$B_7:$	D	E	C	D	B	D	A	
$C_4:$	E	D	D	A				

	1	2	3	4	5	6	7	8
$A_7:$	C	B	E	A	D	D	E	A
$B_6:$	D	E	C	D	B	D	A	
$C_3:$	E	D	D	A				

Nejdelší společná podposloupnost

$$(a_n \neq b_m) \ \& \ (c_k \neq a_n) \implies (C_k = \text{LCS}(A_{n-1}, B_m))$$

	1	2	3	4	5	6	7	8
A₇:	C	B	E	A	D	D	E	

B₆:	D	E	C	D	B	D		
-----------------------	---	---	---	---	---	---	--	--

C₃:	E	D	D					
-----------------------	---	---	---	--	--	--	--	--

	1	2	3	4	5	6	7	8
A₆:	C	B	E	A	D	D	E	

B₆:	D	E	C	D	B	D		
-----------------------	---	---	---	---	---	---	--	--

C₃:	E	D	D					
-----------------------	---	---	---	--	--	--	--	--

$$(a_n \neq b_m) \ \& \ (c_k \neq b_m) \implies (C_k = \text{LCS}(A_n, B_{m-1}))$$

	1	2	3	4	5	6	7	8
A₅:	C	B	E	A	D			

B₅:	D	E	C	D	B			
-----------------------	---	---	---	---	---	--	--	--

C₂:	E	D						
-----------------------	---	---	--	--	--	--	--	--

	1	2	3	4	5	6	7	8
A₅:	C	B	E	A	D			

B₄:	D	E	C	D	B			
-----------------------	---	---	---	---	---	--	--	--

C₂:	E	D						
-----------------------	---	---	--	--	--	--	--	--

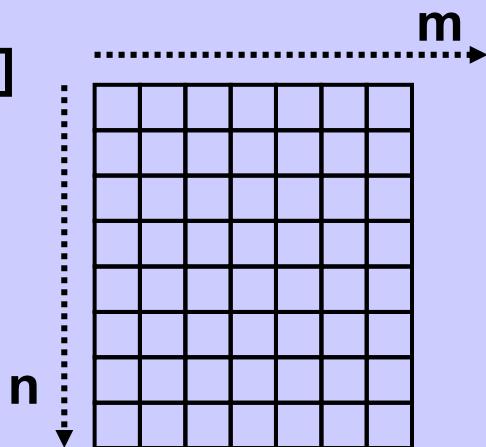
Nejdelší společná podposloupnost

Rekurzivní funkce $C(n,m)$ – délka LCS

$$C(n,m) = \begin{cases} 0 & n = 0 \text{ or } m = 0 \\ C(n-1, m-1) + 1 & n > 0, m > 0, a_n = b_m \\ \max\{ C(n-1, m), C(n, m-1) \} & n > 0, m > 0, a_n \neq b_m \end{cases}$$

Strategie dynamického programování

$C[n][m]$



```
for( a=1; a<=n; a++ )  
  for( b=1; b<=m; b++ )  
    C[a][b] = ... ;  
}
```

Nejdelší společná podposloupnost

Konstrukce DP tabulek pro LCS

```
void findLCS() {  
    for( int a=1; a<=n; a++ )  
        for( int b=1; b<=m; b++ )  
            if( A[a] == B[b] ) {  
                C[a][b] = C[a-1][b-1]+1;  
                arrows[a][b] = DIAG; ↖  
            }  
            else  
                if( C[a-1][b] > C[a][b-1] ) {  
                    C[a][b] = C[a-1][b];  
                    arrows[a][b] = UP; ↑  
                }  
                else {  
                    C[a][b] = C[a][b-1];  
                    arrows[a][b] = LEFT; ←  
                }  
            }  
}
```


Nejdelší společná podposloupnost

Pole
NSP
pro
“CBEADDEA”
a
“DECDBDA”

		C								
		0	1	2	3	4	5	6	7	
		B: D E C D B D A								
A:	0	0	0	0	0	0	0	0	0	
	1	C	0	← ₀	← ₀	↖ ₁	← ₁	← ₁	← ₁	← ₁
	2	B	0	← ₀	← ₀	↑ ₁	← ₁	↖ ₂	← ₂	← ₂
	3	E	0	← ₀	↗ ₁	← ₁	← ₁	↑ ₂	← ₂	← ₂
	4	A	0	← ₀	↑ ₁	← ₁	← ₁	↑ ₂	← ₂	↖ ₃
	5	D	0	↖ ₁	← ₁	← ₁	↗ ₂	← ₂	↖ ₃	← ₃
	6	D	0	↖ ₁	← ₁	← ₁	↖ ₂	← ₂	↗ ₃	← ₃
	7	E	0	↑ ₁	↖ ₂	← ₂	← ₂	← ₂	↑ ₃	← ₃
	8	A	0	↑ ₁	↑ ₂	← ₂	← ₂	← ₂	↑ ₃	↖ ₄

Nejdelší společná podposloupnost

Výpis NSP -- rekurzivně :)

```
void outLCS( int a, int b ) {  
if( a == 0 || b == 0 ) return;  
  
if( arrows[a][b] == DIAG ) {  
    outLCS(a-1, b-1);      // recursion ...  
    print(A[a]);          // ... reverses the sequence!  
}  
else  
    if( arrows[a][b] == UP )  
        outLCS(a-1,b);  
    else  
        outLCS(a,b-1);  
}
```