Subsequência comum máxima

CLRS 15.4

= "recursão–com–tabela"
= transformação inteligente de recursão em iteração

Subsequências

z[0:k] é **subsequência** de x[0:m] se existem índices $i_0 < \cdots < i_{k-1}$ tais que

$$z[0] = x[i_0] \quad \ldots \quad z[k-1] = x[i_{k-1}]$$

EXEMPLOS:

5 9 2 7 é subseq de 9 **5** 6 **9** 6 **2** **7** 3

A A D A A é subseq de A B R A C A D A B R A

```
A           A           D   A           A
|           |           |   |           |
A   B   R   A   C   A   D   A   B   R   A
```

Exercício

Problema: Decidir se z[0:m] é subsequência de x[0:n]

Exercício

Problema: Decidir se z[0:m] é subsequência de x[0:n]

```
def sub_seq (z, x):
   i = len(z)-1
   j = len(x)-1
   while i >= 0 and j >= 0:
       if z[i] == x[j]:
           i -= 1
       j-= 1
   return i< 0
```

Exercício

Problema: Decidir se z[0:m] é subsequência de x[0:n]

```
def sub_seq (z, x):
   i = len(z)-1
   j = len(x)-1
   while i >= 0 and j >= 0:
       if z[i] == x[j]:
           i -= 1
       j-= 1
   return i< 0
```

Consumo de tempo é $O(m + n)$

Subsequência comum máxima

z é subseq comum de x e y

se z é subsequência de x e de y

ssco = subsequência comum

Exemplos: x = A B C B D A B
y = B D C A B A
ssco = B C A
Outra ssco = B D A B

Problema

Problema: Encontrar uma ssco máxima de x e y.

Exemplos: x = A B C B D A B
y = B D C A B A
ssco = B C A
ssco maximal = A B A
ssco máxima = B C A B
Outra ssco máxima = B D A B

LCS = Longest Common Subsequence

diff

```
> more abracadabra     > more yabbadabbadoo
A                      Y
B                      A
R                      B
A                      B
C                      A
A                      D
D                      A
A                      B
B                      B
R                      A
A                      D
                       O
                       O
```

diff -u

```
+Y
 A
 B
-R
-A
-C
+B
 A
 D
 A
 B
-R
+B
 A
+D
+O
+O
```

Subestrutura ótima

Suponha que z[0 :k] é ssco máxima de x[1:m] e y[0 :n].

- Se x[m-1] = y[n-1], então $z[k-1] = x[m-1] = y[n-1]$ e z[0 :k-1] é ssco máxima de x[0:m-1] e y[0:n-1].
- Se $x[m-1] \neq y[n-1]$, então $z[k-1] \neq x[m-1]$ implica que z[0 :k] é ssco máxima de x[0:m-1] e y[0:n].
- Se $x[m-1] \neq y[n-1]$, então $z[k-1] \neq y[n-1]$ implica que z[0 :k] é ssco máxima de x[0:m] e y[0:n-1].

Função recursiva

Retorna o comprimento de uma ssco máxima de x[0:i] e y[0:j].

```
def lcs_rec(x, i, y, j):
    if i == 0 or j == 0:
        return 0
    if x[i-1] == y[j-1]:
        return lcs_rec(x, i-1, y, j-1) + 1
    a = lcs_rec(x, i-1, y, j)
    b = lcs_rec(x, i, y, j-1)
    return max(a, b)
```

Consumo de tempo

$T(m, n) :=$
número máximo de comparações feitas por lcs_rec(x, m, y, n)

Recorrência

$$T(0, n) = 0$$
$$T(m, 0) = 0$$
$$T(m, n) = T(m-1, n) + T(m, n-1) + 1$$
para $n \geq 0$ e $m \geq 0$

$T(m, n)$ é exponencial

Conclusão

O consumo de tempo do algoritmo `lcs_rec()` é exponencial.

Programação dinâmica

Problema: encontrar o comprimento de uma ssco máxima.

$c[i][j] =$ comprimento de uma ssco máxima de $x[0{:}i]$ e $y[0{:}j]$

Recorrência:

$c[0][j] = c[i][0] = 0$

$c[i][j] = c[i-1][j-1] + 1$ se $x[i-1] = y[j-1]$

$c[i][j] = \max(c[i][j-1], c[i-1][j])$ se $x[i-1] \neq y[j-1]$

Programação dinâmica

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

Fórmula fechada

Prove que

$$T(m,n) = \binom{m+n}{m} - 1 .$$

Logo,

$$T(m,m) = \binom{2m}{m} - 1 > \frac{4^m}{2m+1} - 1.$$

Portanto, $T(m,m) > 4^m/m$.

Programação dinâmica

Cada subproblema, comprimento de uma ssco máxima de $x[0{:}i]$ e $y[1{:}j]$, é resolvido uma só vez.

Em que ordem calcular os componentes da tabela c?

Para calcular $c[3,5]$ preciso de $c[3,4]$, $c[2,5]$ e de $c[2,4]$.

Calcule todos os $c[i,j]$ com $i = 1$, $j = 0,1,\ldots,n$,
depois todos com $i = 2$, $j = 0,1,\ldots,n$,
depois todos com $i = 3$, $j = 0,1,\ldots,n$,
etc.

Simulação

	Y		B	D	C	A	B	A
X		0	1	2	3	4	5	6 (j)
	0	0	0	0	0	0	0	0
A	1	0	??					
B	2	0						
C	3	0						
B	4	0						
D	5	0						
A	6	0						
B	7	0						

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Simulação

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

Função de prog-din

Retorna o comprimento de uma ssco máxima de x[0:m] e y[0:n].

```
def lcs_prog_din(x, m, y, n):
  c = crie_matriz(m+1,n+1)
  for i in range(m+1): c[i][0] = 0
  for j in range(n+1): c[0][j] = 0
  for i in range(1,m+1):
     for j in range(1,n+1):
        if x[i-1] == y[j-1]:
           c[i][j] = c[i-1][j-1] + 1
        else:
           c[i][j] = max(c[i-1][j],c[i][j-1])
  return c[m][n]
```

Subsequência comum máxima

Y		B	D	C	A	B	A	
X	0	1	2	3	4	5	6	j
0	⋆	⋆	⋆	⋆	⋆	⋆	⋆	
A 1	⋆	↑	↑	↑	↖	←	↖	
B 2	⋆	↖	←	←	↑	↖	←	
C 3	⋆	↑	↑	↖	←	↑	↑	
B 4	⋆	↖	↑	↑	↑	↖	←	
D 5	⋆	↑	↖	↑	↑	↑	↑	
A 6	⋆	↑	↑	↑	↖	↑	↖	
B 7	⋆	↖	↑	↑	↑	↖	↑	

get_lcs()

```
def get_lcs(x, b):
  z = []
  i = len(b)
  j = len(b[0])
  while i > 0 and j > 0:
     if b[i][j] == "↖":
        z.insert(0, x[i])
        i -= 1
        j -= 1
     elif b[i][j] == "←":
        j -= 1
     else:
        i -= 1
  return z
```

Conclusão

O consumo de tempo da função `lcs_prog_din()` é $O(mn)$.

Função de prog-din

```
def lcs_prog_din(x, m, y, n):
  [...]
  for i in range(1,m+1):
     for j in range(1,n+1):
        if x[i-1] == y[j-1]:
           c[i][j] = c[i-1][j-1] + 1
           b[i][j] = "↖"
        elif c[i-1][j] >= c[i][j-1]:
           c[i][j] = c[i-1][j]
           b[i][j] = "↑"
        else:
           c[i][j] = c[i][j-1]
           b[i][j] = "←"
  return c[m][n]
```