[05/11/2024] → Algoritmi

LCS → Longest Common Subsequence

(SOTTOSEQUENZA PIÙ LUNGA)

✓ RICORRENZA SUI COSTI

$l(i,j) = |LCS(X_i, Y_j)|$

$$l(i,j) = \begin{cases} 0 & \text{se } i = 0 \text{ o } j = 0 \quad \text{(caso 0)} \\ l(i-1, j-1) + 1 & \text{se } i,j > 0 \text{ e } x_i = x_j \quad \text{(caso 1)} \\ \max\{l(i, j-1), l(i-1, j)\} & \text{se } i,j > 0 \text{ e } x_i \neq x_j \quad \text{(caso 2)} \end{cases}$$

ABC
BCD

Alla fine ci interessa calcolare $l(m,n)$.
Per calcolare $l(i,j)$ mi possono servire tre valori:

$$L = \begin{bmatrix} (i-1, j-1) & (i-1, j) \\ \nwarrow & \uparrow \\ (i, j-1) \leftarrow & (i,j) \end{bmatrix}$$

→ MATRICE

Scansione "row-major": riempio la tabella per righe, da sinistra a destra.

Informazione addizionale per costruire la sequenza (vera e propria):

$$b(i,j) = \begin{cases} '\nwarrow' & \text{se } x_i = y_j \\ '\leftarrow' & \text{se } x_i \neq x_j \text{ e } max = LCS(i, j-1) \\ '\uparrow' & \text{se } x_i \neq y_j \text{ e } max = LCS(i-1, j) \end{cases}$$

CAL. SOL.

LCS(X, Y)
```
m = X.length
n = Y.length
for i = 0 to m
    L[i,0] = 0
for j = 0 to n
    L[0,j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i = y_j
            L[i,j] = L[i-1,j-1] + 1
            B[i,j] = '↖'
        else if L[i-1,j] ≥ L[i,j-1]
            L[i,j] = L[i-1,j]
            B[i,j] = '↑'
        else
            L[i,j] = L[i,j-1]
            B[i,j] = '←'
return (L[m,n], B)
```

INIZ.

Domanda B (6 punti) Si calcoli la lunghezza della longest common subsequence (LCS) tra le stringhe *armo* e *toro*, calcolando tutta la tabella $L[i,j]$ delle lunghezze delle LCS sui prefissi usando l'algoritmo visto in classe.

i=0, j=0 →

		T	O	R	O
	0	0	0	0	0
A	0	0 ←	0	0	0
R	0	0	0	1	1
M	0	0	0	1	1
O	0	0	1	1	2

↑ ←

↖

↑ ←

↑ ↖

LUNGHEZZA LCS = 2

↖ = MATCH

↑ , ← = ESISTEVA UN MATCH VICINO

Restituisci $LCS(X,Y)$ e $|LCS(X,Y)|$

$X = \langle b, d, c, d \rangle$
$Y = \langle a, b, c, b, d \rangle$

$$L = \begin{array}{c} \\ b \\ d \\ c \\ d \end{array} \overset{\begin{array}{cccccc} & a & b & c & b & d \end{array}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 2 \\ 0 & 0 & 1 & \mathbf{2} & 2 & 2 \\ 0 & 0 & 1 & 2 & 2 & \mathbf{3} \end{bmatrix}} \qquad B = \begin{array}{c} b \\ d \\ c \\ d \end{array} \overset{\begin{array}{ccccc} a & b & c & b & d \end{array}}{\begin{bmatrix} \uparrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow \\ \uparrow & \uparrow & \uparrow & \uparrow & \nwarrow \\ \uparrow & \uparrow & \nwarrow & \leftarrow & \uparrow \\ \uparrow & \uparrow & \uparrow & \uparrow & \nwarrow \end{bmatrix}}$$

TROVIAMO LA SOLUZIONE

$LCS(X,Y) = \langle b, c, d \rangle \qquad |LCS(X,Y)| = 3$

$\nwarrow$ = HO GIÀ AVUTO IL MATCH

$\leftarrow$ (MATCH NELLA RIGA PER ALTRA COLONNA)

TOP-DOWN

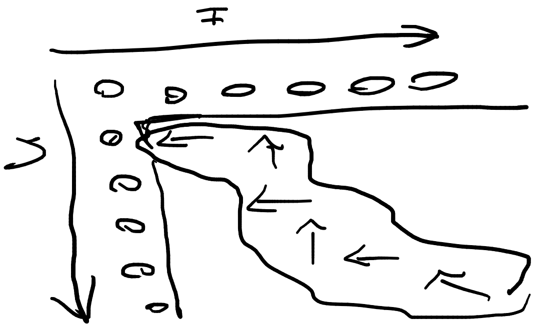
BOTTOM-UP

(CALCOLO OTTIMIZZATO DA SOPRA)

" " DA SOTTO

PROG. DINAMICA
- BOTTOM-UP
- MEMOIZZATO (RICORSIONE)

```
INIT-LCS(X,Y)
  m = X.length
  n = Y.length
  if (m = 0) or (n = 0)
      return 0
  for i = 0 to m
      L[i,0] = 0
  for j = 0 to n
      L[0,j] = 0
  for i = 1 to m
      for i = 1 to n
          L[i,j] = -1
  return R-LCS(X,Y,m,n)
```



MEMOIZATION = CALCOLO DI CORREDO SOLUZIONE

```
R-LCS(X,Y,i,j)
  if L[i,j] = -1
      if x_i = y_j
          L[i,j] = R-LCS(X,Y,i-1,j-1)
      else if R-LCS(X,Y,i-1,j) ≥ R-LCS(X,Y,i,j-1)
          L[i,j] = L[i-1,j]
      else
          L[i,j] = L[i,j-1]
  return L[i,j]
```

Esercizio 2 (9 punti) Data una stringa $X = x_1, x_2, \ldots, x_n$, si consideri la seguente quantità $\ell(i,j)$, definita per $1 \leq i \leq j \leq n$:

$$\ell(i,j) = \begin{cases} 1 & \text{se } i = j \\ 2 & \text{se } i = j-1 \\ 2 + \ell(i+1, j-1) & \text{se } (i < j-1) \text{ e } (x_i = x_j) \\ \sum_{k=i}^{j-1} (\ell(i,k) + \ell(k+1,j)) & \text{se } (i < j-1) \text{ e } (x_i \neq x_j). \end{cases}$$

1. Scrivere una coppia di algoritmi INIT_L(X) e REC_L(X, i, j) per il calcolo memoizzato di $\ell(1, n)$.
2. Si determini la complessità *al caso migliore* $T_{\text{best}}(n)$, supponendo che le uniche operazioni di costo unitario e non nullo siano i confronti tra caratteri.

LUNGHEZZA ARRAY

DOVE MATRICE [i,j] = 0 → INSERISCO SOLUZIONE

```
INIT_L(X)
 n <- length(X)
 if n = 1 then return 1
 if n = 2 then return 2
 for i=1 to n-1 do
   L[i,i] <- 1
   L[i,i+1] <- 2
 L[n,n] <- 1
 for i=1 to n-2 do
   for j=i+2 to n do
     L[i,j] <- 0
 return REC_L(X,1,n)
```

n = 5

i = j-1

i = j

j = i+1

L[i,j] = 1

L[i,i+1]

```
REC_L(X,i,j)
 if L[i,j] = 0 then
   if x_i = x_j then L[i,j] <- 2 + REC_L(X,i+1,j-1)
   else for k=i to j-1 do
          L[i,j] <- L[i,j] + REC_L(X,i,k) + REC_L(X,k+1,j)
 return L[i,j]
```

Esercizio 2 (9 punti) Sia n un intero positivo. Si consideri la seguente ricorrenza $M(i,j)$ definita su tutte le coppie (i,j) con $1 \le i \le j \le n$:

$$M(i,j)=\begin{cases}2 & \text{se } i=j,\\ 3 & \text{se } j=i+1,\\ M(i+1,j-1)\cdot M(i+1,j)+M(i,j-1) & \text{se } j>i+1.\end{cases}$$

(caso base)

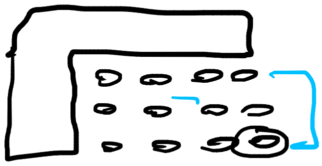
(vedo (1,n) e so che la scansione è per diagonale)

(a) Si scriva una coppia di algoritmi INIT_M(n) e REC_M(i,j) per il calcolo memoizzato di $M(1,n)$. → RICORSIONE

(b) Si calcoli il numero esatto $T(n)$ di moltiplicazioni tra interi eseguite per il calcolo di $M(1,n)$.

REC_M (i,j)

IF M[i,j]=0



→ SOLUZIONE

M[i,j] = REC_M(i+1, j−1) ·
REC_M(i+1, j) + REC_M(i, j−1) ← PEZZO IN GIALLO

RETURN M(i,j)

RICORRENZA → { M(i,j) → 2 i=j ; 3 j=i+1 }

INIT_M (n) ← n = LENGTH OF ARRAY

M(i,j) → (i,j) ↑ ∃ n

1 ≤ i ≤ j ≤ n

{ i=j → n=1 → ESISTE 1 SOLO ELEMENTO }

IF n=1 THEN RETURN 2

IF n=2 THEN RETURN 3

[j=i+1, n=2]

[2 → se i=j]

FOR i=1 TO N−1 DO

M[i,i] = 2 →(i=j)

M[i,i+1] = 3 (j=i+1)

// M(1,n) → M(n,n)=2 3

[ ] ← 2

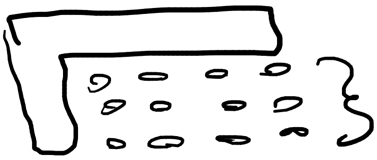
// FOR I = 1 TO [N - 1] → GIÀ RIEMPITO

// [J = I + 1]

FOR I = 1 TO N - 2 DO

FOR J = I + 2 TO N DO

M[I, J] = 0 → RIEMPITO PER SAPERE CHE LI CALCOLIAMO SOLUZIONE



SOL.

```
(a) INIT_M(n)
    if n=1 then return 2
    if n=2 then return 3
    for i=1 to n-1 do
        M[i,i] = 2
        M[i,i+1] = 3
    M[n,n] = 2
    for i=1 to n-2 do
        for j=i+2 to n do
            M[i,j] = 0
    return REC_M(1,n)
```

Ultimo caso dell'indice [i, i]

Se gli indici sono uno solo o due soli, allora, seguendo i casi base si ritorna 2 e 3.

Ricordandosi che:
- "i" va da i ~~ad (n-1)~~, quindi diventa perché abbiamo già inizializzato, (n-2) la scansione
- "j" va ~~da (n-1)~~ a 0 compresi, quindi dato che abbiamo inizializzato, parte da (n-2)

Una volta inizializzato secondo i casi base, tutto il resto della matrice viene riempita di zeri. Poi, si considera, essendo una scansione per diagonale, noi partiamo dal basso e riempiamo solo la parte sopra delle diagonali principali, ovviamente riempita anche questa di zeri.

SOMMATORIE → CALCOLO ESATTO N. OPERAZIONI

$$\underbrace{T(n) = T\left(\frac{n}{2}\right) + c}_{} \;\dots\to (?)$$

$1^0 \to 7$

```
REC_M(i,j)
if M[i,j] = 0 then
   M[i,j] = REC_M(i+1,j-1) * REC_M(i+1,j) + REC_M(i,j-1)
return M[i,j]
```

$$\sum_{i=1}^{n-2} \sum_{j=i+2}^{n} (1)$$

1 MOLTIPLICAZIONE

$$A \overset{1}{\otimes} B \overset{2}{\otimes} C \sim \sum_i \sum_j (2)$$

$$A \otimes B$$

....

2 SOMMATORIE 1

$$(\sum\sum) \to \sum_{i=1}^{n-2} [n - i - 1]$$

(GAUSS)

$$\left[\frac{(n-2)(n-1)}{2}\right] \longleftrightarrow k = \frac{n(n-1)}{2}$$

$$\sum (k)$$