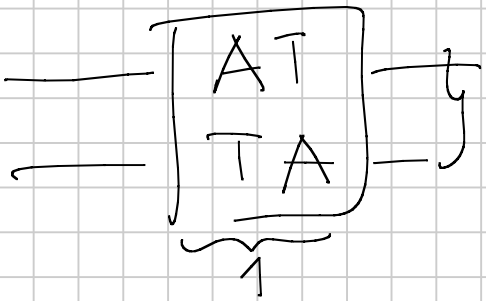
ESERCITAZIONE: Programmazione Dinamica

Titolo nota 26/02/2016

ESERCIZIO 1



Si consideri il problema dell'ED. e si ammetta che tra i possibili errori ci sia anche l'INVERSIONE tra due caratteri adiacenti, e che tale errore abbia peso 1.

1) Mostrare la nuova regola ricorsiva che include anche l'inversione
2) Mostrare la matrice di PD relativa alle sequenze ATLETA e ALTERA
3) Indicare l'allineamento (o gli allineamenti) ottimo delle due sequenze

A T L E T A
A L T E R A

distanza 2

① Regola ricorsiva

$$M[i,j] = \text{distanza}(X_i, Y_j) = \begin{cases} i & j = 0 \\ j & i = 0 \\ \min\{M[i-1,j]+1,\ M[i-1,j-1]+P,\ M[i,j-1]+1\} & i>0,\ j>0 \\ \min\{M[i-1,j]+1,\ M[i-1,j-1]+P,\ M[i,j-1]+1,\ M[i-2,j-2]+1\} & \end{cases}$$

$$P = \begin{cases} 1 & x_i \neq y_j \\ 0 & x_i = y_j \end{cases}$$

se non c'è inversione

se invece

- $i>1,\ j>1$
  $x_i = y_{j-1}$
  $y_j = x_{i-1}$
  } inversione

X_{i-2} | X_{i-1} x_i
Y_{j-2} | y_{j-1} y_j

$M[i-2, j-2]$ 1

Titolo nota 26/02/2016

	∅	A	L	T	E	R	A
∅	0	1	2	3	4	5	6
A	1	0	1	2	3	4	5
T	2	1	1	1	2	3	4
L	3	2	1	1	2	3	4
E	4	3	2	2	1	2	3
T	5	4	3	2	2	2	3
A	6	5	4	3	3	3	2

1

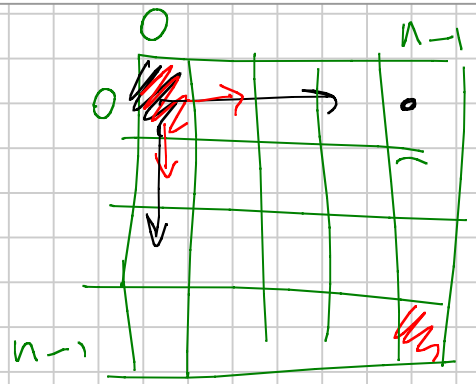
ATLETA

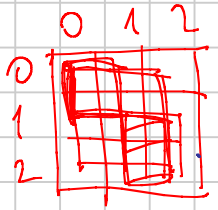
ALTERA

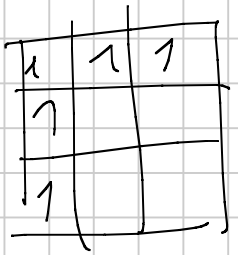
0 1 0 1 0

2

ESERCIZIO 1







① Sottoproblemi

P $(0,0) \longrightarrow (n-1, n-1)$

P_{ij} = $(0,0) \longrightarrow (i,j)$ $\quad 0 \le i \le n-1$, $0 \le j \le n-1$

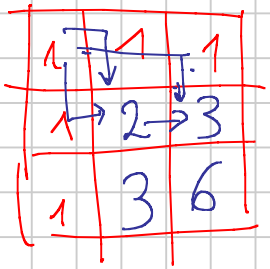
M $n \times n$ $\quad M(i,j) = \#$ percorsi $(0,0) \to (i,j)$

Titolo nota 26/02/2016

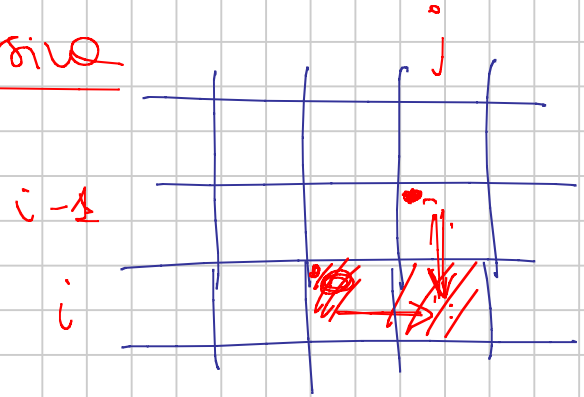
② Sottoproblemi elementari

$M(i,0) = M[0,j] = 1$

$0 \le i \le n-1$
$0 \le j \le n-1$



③ Regola ricorsiva



$M(i,j) = M(i-1,j) + M(i,j-1)$

$i > 0$
$j > 0$

④ return $M(n-1, n-1)$

ContaPercorsi (n)

```
M = nuova matrice n×n
// sottoproblemi elementari
for i = 0 to n-1
        M[i,0] = 1      // prima colonna
for j = 1 to n-1
        M[0,j] = 1      // prima riga
// riempimento matrice
for i = 1 to n-1 {
   for j = 1 to n-1 {
        M[i,j] = M[i-1,j] + M[i,j-1];
   }
}
return M[n-1, n-1]
```

$T(n) = \Theta(n^2)$

ESERCIZIO 2

Titolo nota 26/02/2016

① Sottoproblemi

$(0,0) \longrightarrow (i,j)$

$(0,0) \longrightarrow (n-1, n-1)$

massimizzando il profitto

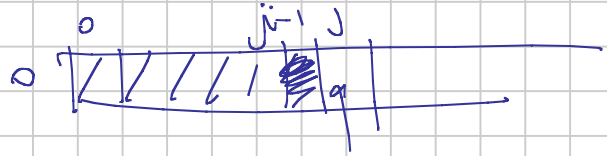
$0 \le i \le n-1$

$0 \le j \le n-1$

T $n \times n$

② Sottoproblemi elementari

$(0,0) \longrightarrow (0,j)$



$T(i,j)$ = valore (del comune di valore) massimo dei cammini $(0,0) \to (i,j)$

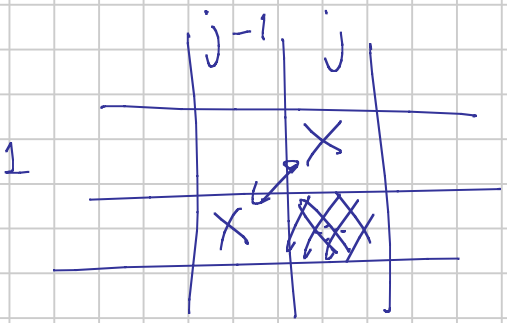
$j>0$: $T[0,j] = T[0,j-1] + C_{0j}$

$(0,0) \longrightarrow (i,0)$

$i>0$ $T[i,0] = T[i-1,0] + C_{i0}$

C_{ij} = valore della cella (i,j)

③ Regola ricorsiva



$$T[i,j] = \max\{T[i-1,j], T[i,j-1]\} + C_{ij}$$

matrice

④ return $T[n-1, n-1]$

Scacchiera (C) // C: matrice dei valori delle caselle

```
T = nuova matrice n×n
T[0,0] = C[0,0]
for i = 1 to n-1 { T[i,0] = T[i-1,0] + C[i,0] }   // prima colonna
for j = 1 to n-1 { T[0,j] = T[0,j-1] + C[0,j] }   // prima riga
for i = 1 to n-1 {
   for j = 1 to n-1 {
        if ( T[i-1,j] ≥ T[i,j-1] )  T[i,j] = T[i-1,j] + C[i,j];
        else T[i,j] = T[i,j-1] + C[i,j];
} }
// ricostruzione del cammino
cammino = nuovo array di dim 2n-2   // il cammino è di lunghezza 2n-2
i = n-1
j = n-1
```

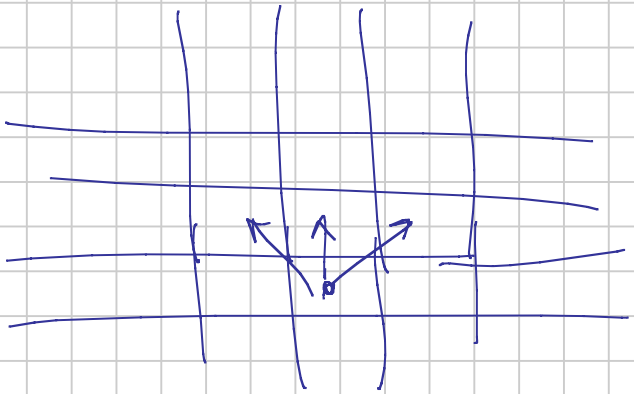
↓ →

```
for k = 2n-3 downto 0 {
    if ( T(i,j) = C(i,j) + T(i-1, j) ) {
            comune(k) = '↓'
            i--;
    }
    else {  comune[k] = '→'
            j--;
    }
}
print (comune);
return T(n-1, n-1);
```

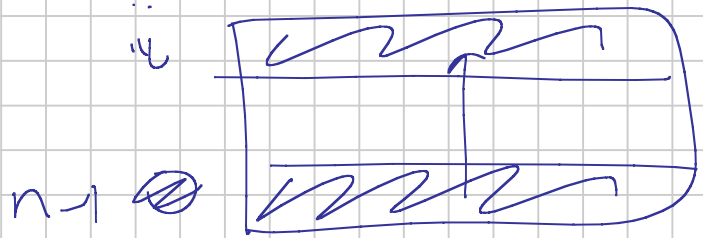
$T(n) = \Theta(n^2)$

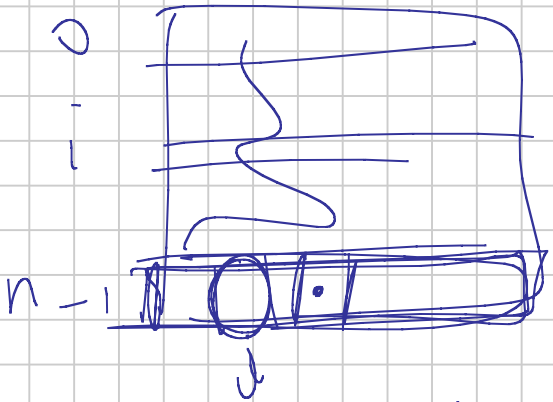
Titolo nota 26/02/2016

ESERCIZIO 3



(1)

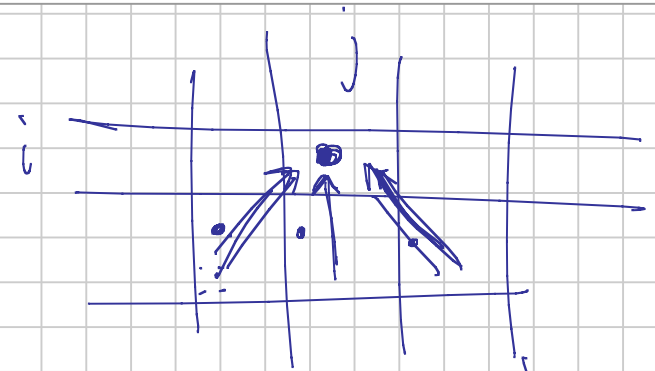




$q[n-1, j] = p(n-1, j)$

q = matrice PD
p = ~~int~~ matrice profitti

Regola ricorsiva



$$g(i,j) = \max \begin{cases} g(i+1, j-1) & j>0 \\ g(i+1, j) & \\ g(i+1, j+1) & j<n-1 \end{cases} + p(i,j)$$

(4)

Soluzione

max della prima riga di g

$$\max \{ g(0,j) \mid 0 \le j \le n-1 \}$$

Matrice per la ricostruzione del percorso


```
RicercaGamerino(p)   // p: matrice dei profitti
① { q = nuova matrice n×n      // matrice PD, max profitto del corriere
    m = nuova matrice n×n      // matrice per la ricostruzione del percorso
    // l'ultima riga di Q si inizializza con i profitti
② { for j=0 to n-1
        q[n-1,j] = p[n-1,j]
    // riempimento della tabella dal basso verso l'alto
③ for i = n-2 downto 0 {
      for j = 0 to n-1 {
```

```
// ricerca del massimo tra g[i+1,j-1], g[i+1,j], g[i+1,j+1]
g[i,j] = g[i+1,j];
m[i,j] = 0                                   //  ↑
if (j>0 && g[i+1,j-1] > g[i,j]) {
        g[i,j] = g[i+1,j-1];
        m[i,j] = -1;                         //  ↗
}
if (j<n-1 && g[i+1,j+1] > g[i,j]) {
        g[i,j] = g[i+1,j+1];
        m[i,j] = +1                          //  ↖
}
g[i,j] = g[i,j] + p[i,j];
}
}
// ricerca della casella iniziale col massimo profitto
max = g[0,0];
start = 0;
```

```
for j = 1 to n-1 {
    if (q[0,j] > max) {
        max = q[0,j];
        start = j;
    }
}
// max = massimo profitto, j = indice della cella nella prima riga col max profitto
// stampa del percorso
j = start
for i = 0 to n-2 {
    print "(i,j) → (i+1, j + m[i,j])"
    j = j + m[i,j];
}
```

$T(n) = \Theta(n^2)$