**Algoritmica 07/08**
**13. PROGRAMMAZIONE DINAMICA**

Per un'esposizione generale di questo paradigma algoritmico si veda il testo B, Introduzione al cap. 15 e par. 15.3.
Per l'applicazione del paradigma al **prodotto di una catena di matrici** si veda il testo B, par. 15.2. È un esempio importante.

La programmazione dinamica si usa nei casi in cui esista una definizione ricorsiva del problema, ma la trasformazione diretta di tale definizione in un algoritmo genera un programma di complessità esponenziale a causa del calcolo ripetuto sugli stessi sottoinsiemi di dati da parte delle diverse chiamate ricorsive.

Due esempi elementari sono il calcolo dei numeri di Fibonacci e dei coefficienti binomiali. Si ha rispettivamente:

$F_i = F_{i-1} + F_{i-2}$ $\qquad$ (n su k) = (n-1 su k) + (n-1 su k-1)

Entrambe le formule conducono a un numero esponenziale di operazioni se interpretate come espressioni di calcolo ricorsivo (per es. calcolare $F_i$ chiamando ricorsivamente la stessa procedura per calcolare $F_{i-1}$ e $F_{i-2}$). Verificare personalmente questo punto.
Se invece, partendo da $F_0=0$, $F_1=1$, si calcolano i numeri di Fibonacci in sequenza come $F_2=F_1+F_0$, $F_3=...$, si arriva a $F_i$ in i-1 passi e l'algoritmo è quindi lineare (a parte il problema, che esiste in ogni caso, della crescita esponenziale del *valore* dei numeri nella sequenza).
Similmente si può calcolare (n su k) riempiendo le prime n righe del triangolo di Tartaglia a partire dalla riga 0 fino alla riga n-1, calcolando gli elementi della riga i per addizioni sulla riga i-1. Si può quindi calcolare direttamente il valore (n su k)=(n-1 su k)+(n-1 su k-1) con un'addizione perché i due addendi nella riga n-1 sono ora noti. Il numero di passi è quadratico, perché le caselle del triangolo fino alla riga n-1 sono circa $n^2/2$.

Applicazioni classiche della programmazione dinamica si incontrano nel **confronto tra sequenze di caratteri**: problemi nati in relazione agli editori di testo, e oggi importantissimi nelle applicazioni algoritmiche in biologia molecolare (analisi del DNA ecc). Vediamo lo schema di base e come questo possa essere facilmente modificato per risolvere diversi problemi.

**Edit distance.** Date due sequenze di caratteri X,Y trovare un *allineamento ottimo* delle due che corrisponda alla *minima distanza* tra X e Y (vedi sotto). Si ammette che nelle due sequenze possano essere inseriti spazi (indicati con -) e che i caratteri di una corrispondano a caratteri o spazi nell'altra. La distanza è la somma delle distanze tra coppie di caratteri o spazi, uno in X e l'altro in Y, in posizioni corrispondenti nell'allineamento: caratteri uguali hanno distanza Ø (**match**); caratteri diversi hanno

distanza 1 (**mismatch**), carattere allineato a spazio hanno distanza 1. Si noti che la presenza di uno spazio si interpreta come l'avvenuta **cancellazione** di un carattere dalla sequenza, o come l'**inserzione** di un carattere nell'altra.

Valga come esempio la distanza tra le sequenze alfabetiche X = ALBERO e Y = LABBRO. Due allineamenti ottimi con distanza =3, che è la minima possibile (i segni + indicano i caratteri diversi nell'allineamento) sono i seguenti:

```
A L B E R O        A L - B E R O
L A B B R O        - L A B B R O
+ +   +            +   +   +
```

Dette $X = x_1\, x_2 \ldots x_n$ , $Y = y_1\, y_2 \ldots y_m$ , il calcolo impiega una matrice M di dimensioni $(n+1)\times(m+1)$ ove M[i,j] indica la distanza tra il prefisso $x_1\, x_2 \ldots x_i$ di X e il prefisso $y_1\, y_2 \ldots y_j$ di Y: dunque i caratteri di X corrispondono alle righe di M, i caratteri di Y alle colonne, e l'**ultimo valore** M[n,m] indica la **distanza tra le due sequenze**. La riga e la colonna Ø corrispondono a prefissi vuoti, cioè X e Y non sono ancora stati esaminati, quindi la M si inizializza sulla riga Ø e la colonna Ø ponendo:

$M[Ø,j]=j$ per $Ø \le j \le m$, $M[i,Ø]=i$ per $Ø \le i \le n$,

valori che corrispondono ad affermare che il prefisso vuoto di X allineato con il prefisso $y_1\, y_2 \ldots y_j$ di Y corrisponde a una distanza j, e che il prefisso vuoto di Y allineato con il prefisso $x_1\, x_2 \ldots x_i$ di X corrisponde a una distanza i. Per esempio M[Ø,3] = 3 corrisponde all'allineamento del prefisso LAB con ---.

Posto $p(i,j)=Ø$ se $x_i=y_j$ (match), $p(i,j)=1$ se $x_i \neq y_j$ (mismatch), gli elementi M[i,j] con $1 \le i \le n$ e $1 \le j \le m$ si calcolano progressivamente, riga per riga, con la formula ricorsiva:

$$M[i,j] = \min\{M[i,j-1]+1,\ M[i-1,j]+1,\ M[i-1,j-1]+p(i,j)\}$$

la cui giustificazione dovrebbe essere chiara. Avremo:

M[i,j] =

| | Ø | L | A | B | B | R | O |
|---|---|---|---|---|---|---|---|
| Ø | Ø | 1 | 2 | 3 | 4 | 5 | 6 |
| A | 1 | 1 | 1 | 2 | 3 | 4 | 5 |
| L | 2 | 1 | 2 | 2 | 3 | 4 | 5 |
| B | 3 | 2 | 2 | 2 | 2 | 3 | 4 |
| E | 4 | 3 | 3 | 3 | 3 | 3 | 4 |
| R | 5 | 4 | 4 | 4 | 4 | 3 | 4 |
| O | 6 | 5 | 5 | 5 | 5 | 4 | 3 |

L'ultimo valore M[6,6] = 3 indica la distanza tra le due sequenze. Gli allineamenti che danno luogo a questa distanza si ricostruiscono risalendo all'indietro nella matrice, dalla casella [6,6] fino alla casella [Ø,Ø]. Da M[i,j] si risale a M[i-1,j-1] se questi due valori sono uguali e $x_i=y_j$ (per esempio si risale da M[6,6] a M[5,5] entrambi = 3, che indica un match tra caratteri); oppure si risale al valore di M uguale a M[i,j]-1 tra i tre adiacenti che lo precedono: in questo caso vi possono essere più alternative, corrispondenti ad allineamenti di uguale distanza. Nell'esempio si ricostruiscano i due allineamenti ottimi tra ALBERO e LABBRO già indicati in precedenza, partendo da M[6,6].

L'algoritmo per calcolare la Edit Distance ha dunque **complessità quadratica** $\Theta(nm)$ perché richiede di costruire una matrice M di dimensioni $(n+1)\times(m+1)$, e il valore in ciascuna cella è calcolato in tempo costane con la formula ricorsiva indicata sopra perché i tre valori che vi appaiono sono stati già calcolati in passi precedenti e memorizzati in M. Se si richiede di ricostruire un solo allineamento ottimo, il successivo algoritmo svolge tale compito in tempo $\Theta(n+m)$ per tracciare il percorso all'indietro dalla casella [n,m] alla [Ø,Ø].

A titolo di ulteriore esempio si consideri l'elemento M[3,6]=4 corrispondente al confronto tra ALB e LABBRO cui corrisponde l'allineamento ottimo:

```
- A L B - -
L A B B R O
+   +   + +
```

Oppure l'elemento M[4,3]=3 corrispondente al confronto tra ALBE e LAB cui corrispondono gli allineamenti ottimi:

```
A L B E     A L B E     - A L B E
- L A B     L A B -     L A - B -
+   + +     + +   +     +   +   +
```

Il calcolo di distanza mediante la matrice M può essere facilmente trasformato per risolvere problemi diversi dalla classica Edit Distance. Anzitutto si può considerare lo stesso problema ponendo che il costo di mismatch tra caratteri sia diverso da quello di cancellazione o inserzione, cioè il valore di p(i,j) per $x_i\neq y_j$ nella ricorrenza vista sopra non sia uguale a quello di allineamento tra un carattere e uno spazio. Come esercizio si calcoli nuovamente la distanza tra ALBERO e LABBRO ponendo p(i,j)=2 per $x_i\neq y_j$.

Un altro problema di grande importanza, sia per gli editori di testo che per la biologia molecolare, è quello di trovare tutte le **apparizioni approssimate** di una sequenza corta (la X, detta ora **pattern**) in una sequenza molto più lunga (la Y, detta ora **testo**), ammettendo che queste apparizioni possano contenere qualche differenza, ovvero che la distanza tra la X e la sottosequenza di Y con cui si confronta la X possa non essere zero. A tale scopo si può impiegare lo stesso algoritmo per l'Edit Distance classica, con

l’unica variazione di inizializzare la riga zero con tutti Ø, cioè porre:

$M[Ø,j]=Ø \quad \text{per } Ø \leq j \leq m,$

che corrisponde a iniziare il confronto tra la X e una sottosequenza di Y che parta da qualsiasi posizione j in questa senza che ciò contribuisca al valore della distanza.

La seguente matrice è relativa alla ricerca delle apparizioni di X = RAT in Y = SERRATURA.

| M[i,j] = | | Ø | S | E | R | R | A | T | U | R | A |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø | Ø |
| | R | 1 | 1 | 1 | Ø | Ø | 1 | 1 | 1 | Ø | 1 |
| | A | 2 | 2 | 2 | 1 | 1 | Ø | 1 | 2 | 1 | Ø |
| | T | 3 | 3 | 3 | 2 | 2 | 1 | Ø | 1 | 2 | 1 |

I valori nell’ultima riga indicano la distanza tra la X e le sottosequenze di Y che terminano in ogni sua posizione. Per esempio M[3,6]=Ø corrisponde all’apparizione esatta di RAT nelle posizioni 4,5,6 di SERRATURA.

Per determinare gli allineamenti tra la X e le sottosequenze di Y si procede come nel caso precedente, risalendo dalla posizione considerata nell’ultima riga fino alla prima riga, ma senza dover poi recedere fino alla posizione [Ø,Ø]. Per esempio l’ultimo elemento M[3,9]=1 corrisponde all’allineamento di RAT con RA-. L’elemento M[3,4]=2 corrisponde ai tre allineamenti di RAT con RR-, R-R, R--.