Distanza di Edit

Speaker: Antinisca Di Marco

Data: 14-04-2016

Confronto di sequenze

Il confronto tra sequenze in biologia computazionale è la base per:

- misurare la “similarità” tra le sequenze
  - allineamento
- **misurare la “diversità” tra le sequenze**
  - **distanza di edit**
- trovare parti comuni alle sequenze
  - pattern discovery
  - allineamento locale

Distanza tra due sequenze

Distanza tra due sequenze S_1 e S_2 (Levenshtein 66):

numero minimo di “operazioni di modifica” elementari necessarie per trasformare S_1 in S_2

Per modifica elementare si intende la cancellazione (c) di un carattere, la sostituzione (s) di un carattere con un altro, o l'inserimento (i) di un carattere.

la **distanza di Levenshtein**, o **distanza di edit**, è una misura per la differenza fra due stringhe. Introdotta dallo scienziato russo Vladimir Levenshtein nel 1965, serve a determinare quanto due stringhe siano simili.

Per esempio, per trasformare "bar" in "biro" occorrono due modifiche:
"bar" -> "bir" (sostituzione di 'a' con 'i')
"bir" -> "biro" (inserimento di 'o')
Non è possibile trasformare la prima parola nella seconda con meno di due modifiche, quindi la distanza di Levenshtein fra "bar" e "biro" è 2.

Edit transcript

I = inserisci, C = cancella, S = sostituisci, N = lascia invariato

v intner → SINCNCNNI →

wri t ers

SINCNCNNI Rappresenta una particolare trasformazione di una stringa in un’altra

Distanza di edit: il problema

INPUT:

due sequenze S_1 e S_2 definite su un alfabeto Σ

OUTPUT:

distanza di edit tra S_1 e S_2

edit transcript ottimale che fornisce la trasformazione da S_1 a S_2

TECNICA UTILIZZATA: Programmazione Dinamica (PD)

Programmazione Dinamica

Passi fondamentali della Programmazione Dinamica

- Riduzione del problema in sottoproblemi

- Risoluzione di tutti i sottoproblemi possibili

- Risoluzione del problema originale tramite l'utilizzo delle soluzioni dei suoi sottoproblemi

Sembra Divide et impera…che differenza c'e'?

Introduzione alla programmazione

Calcolo della distanza di edit

Si considerino le sequenze :

$S_1 =$ $a_1\ a_2\ \ldots\ a_{i-1}\ a_i\ a_{i+1}\ \ldots\ a_n$

$S_2 =$ $b_1\ b_2\ \ldots\ldots\ b_{j-1}\ b_j\ b_{j+1}\ldots\ b_m$

Costruiamo l'array:

D(i,j) = distanza tra il prefisso $a_1a_2\ldots a_i$ e il prefisso $b_1b_2\ldots b_j$

Il risultato cercato sarà:

D(n,m) = distanza tra $a_1\ a_2\ \ldots\ a_n$ e $b_1\ b_2\ \ldots b_m$

Calcolo della distanza di edit

Si hanno tre possibilità per calcolare $D(i,j)$, noto $D(k,l)$ per $k<i$ e $l<j$:

$t(a_i,b_j)=1$ se a_i diverso da b_j, altrimenti $t(a_i,b_j)=0$

- ✓ il carattere a_i va sostituito con il co

$$D(i,j) = D(i-1,j-1)+t(a_i,b_j)$$

- ✓ il carattere a_i va cancellato e quindi:

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

- ✓ il carattere b_j va inserito e quindi:

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

Calcolo della distanza di edit

- Si richiama quindi il calcolo di:

D(i-1,j-1)

$a_1\ a_2\ \ldots\ a_{i-1}\ a_i\ a_{i+1}\ \ldots\ a_n$

$b_1\ b_2\ \ldots\ldots\ b_{j-1}\ b_j\ \ldots\ b_m$

D(i-1,j)

$a_1\ a_2\ \ldots\ a_{i-1}\ a_i\ a_{i+1}\ \ldots\ a_n$

$b_1\ b_2\ \ldots\ldots\ b_{j-1}\ b_j\ \ldots\ b_m$

D(i,j-1)

$a_1\ a_2\ \ldots\ a_{i-1}\ a_i\ a_{i+1}\ \ldots\ a_n$

$b_1\ b_2\ \ldots\ldots\ b_{j-1}\ b_j\ \ldots\ b_m$

Calcolo della distanza di edit

Dal momento che si vuole un valore minimo, si ottiene la ricorrenza

$$D(i,j) = \mathrm{MIN} \begin{cases} D(i-1,j-1)+t(a_i,b_j) \\ D(i-1,j)+1 \\ D(i,j-1)+1 \end{cases}$$

che stabilisce un legame tra il generico sottoproblema D(i,j) e i sottoproblemi D(i-1,j-1), D(i-1,j) e D(i,j-1)

Calcolo della distanza di edit

Ricorrenza:

stabilisce un legame ricorsivo tra il valore di D(i,j) e i valori per indici più piccoli, fino a un valore base.

BASE: $D(i,0) = i$ $D(0,j) = j$

PASSO: $D(i,j) = \min\{D(i-1,j)+1;\ D(i,j-1)+1;\ D(i-1,j-1)+t(i,j)\}$

con $t(i,j) = 1$ se $a_i \neq b_j$

0 altrimenti

Calcolo della distanza di edit

In particolare:

- per $i=n$ e $j=m$, si ottiene la distanza di edit $D(n,m)$ tra le sequenze S_1 e S_2
- per $i=0$ e $j>0$, si ottiene la distanza di edit $D(0,j)$ tra la sequenza nulla ε e il prefisso $b_1b_2\ldots b_j$ $(D(0,j)=j)$
- per $i>0$ e $j=0$, si ottiene la distanza di edit $D(i,0)$ tra il prefisso $a_1a_2\ldots a_i$ e la sequenza nulla ε $(D(i,0)=i)$

Calcolo della distanza di edit

I casi base, per i quali il valore di D è calcolabile immediatamente, sono:

- $D(0,0) = 0$
- $D(i,0) = i$ (i cancellazioni)
- $D(0,j) = j$ (j cancellazioni)

Calcolo della distanza di edit

Esempio: calcolo della distanza di edit per S_1="winter" (n=6) e S_2="writers" (m=7)

	ε	w	r	i	t	e	r	s
ε	0	1	2	3	4	5	6	7
w	1	0	1	2	3	4	5	6
i	2	1	1	1	2	3	4	5
n	3	2	2	2	2	3	4	5
t	4	3	3	3	2	3	4	5
e	5	4	4	4	3	2	3	4
r	6	5	4	5	4	3	2	**3**

Nella cella D(6,7) è memorizzata la distanza di edit tra S_1 e S_2

Calcolo della distanza di edit

Esempio: ricostruzione della trasformazione da S_1=“winter” a S_2=“writers”

…e così di seguito

writers

Operazioni:

- inserimento di **s**
- sostituzione **n** → **i**
- sostituzione **i** → **r**

		w	r	i	t	e	r	s
	0	1	2	3	4	5	6	7
w	1	**0**	1	2	3	4	5	6
i	2	1	**1**	1	2	3	4	5
n	3	2	2	**2**	2	3	4	5
t	4	3	3	3	**2**	3	4	5
e	5	4	4	4	3	**2**	3	4
r	6	5	4	5	4	3	**2**	**3**

Calcolo della distanza di edit

La complessità in tempo dell'algoritmo è

- O(nm) per il riempimento della matrice di calcolo della distanza di edit
- O(n+m) per la ricostruzione della trasformazione da S_1 a S_2

Limiti Superiori e Inferiori

La distanza di Levenshtein ha alcuni semplici limiti superiori ed inferiori:

- è almeno la differenza fra le lunghezze delle due stringhe;
- è 0 se e solo se le due stringhe sono identiche;
- se le lunghezze delle due stringhe sono uguali, la distanza di Levenshtein non supera la distanza di Hamming, cioè pari alla lunghezza delle stringhe;
- il limite superiore è pari alla lunghezza della stringa più lunga.