Avventure Algoritmiche

Bioinformatica e Sequenziamento di Genomi - II

Alberto Policriti

Udine, 30.10.2019

Il Piano

Questioni di base

1. cosa significa *allineare* (due o più sequenze, e.g. di DNA)?
2. perché il problema si presta ad illustrare *algoritmi*?
3. quali obiettivi abbiamo?
4. che dati utilizziamo per fare esperimenti?

Allineamento

Il significato (operativo) del termine allineare

Allineare due stringhe significa ...

... convertire la prima nella seconda

Il significato (operativo) del termine allineare

Allineare due stringhe significa ...

... convertire la prima nella seconda

Example

$$\sigma_1 \equiv \quad a \quad c \quad g \quad t \quad c \quad a \quad t \quad c \quad a$$
$$\sigma_2 \equiv \quad t \quad a \quad a \quad g \quad t \quad g \quad t \quad c \quad a$$

Che cosa intendiamo per convertire una sequenza in un'altra?

Definizione

Un allineamento di σ_1 e σ_2 è una stringa nell'alfabeto

$$\{\mathbf{m}, \mathbf{i}, \mathbf{d}, \mathbf{s}\}$$

Example

$\sigma_1 \equiv$		a	c	g	t	c	a	t	c	a
	i	**m**	**s**	**m**	**m**	**s**	**d**	**m**	**m**	**m**
$\sigma_2 \equiv$	t	a	a	g	t	g		t	c	a

Che cosa intendiamo per convertire una sequenza in un'altra?

Definizione

Un allineamento di σ_1 e σ_2 è una stringa nell'alfabeto

$$\{\mathbf{m}, \mathbf{i}, \mathbf{d}, \mathbf{s}\}$$

Example

$\sigma_1 \equiv$		a	c	g	t	c	a	t	c	a
	i	**m**	**s**	**m**	**m**	**s**	**d**	**m**	**m**	**m**
$\sigma_2 \equiv$	t	a	a	g	t	g		t	c	a

Un allineamento è una stringa

allineamento $\Leftrightarrow$ edit-string

In in effetti è un (semplice, lineare) *"programma"* che converte σ_1 in σ_2 (o viceversa).

Che cosa intendiamo per convertire una sequenza in un'altra?

Definizione

Un allineamento di σ_1 e σ_2 è una stringa nell'alfabeto

$$\{\mathbf{m}, \mathbf{i}, \mathbf{d}, \mathbf{s}\}$$

Example

$\sigma_1 \equiv$		a	c	g	t	c	a	t	c	a
	i	**m**	**s**	**m**	**m**	**s**	**d**	**m**	**m**	**m**
$\sigma_2 \equiv$	t	a	a	g	t	g		t	c	a

Due problemi

1. trovare un allineamento
2. stabilire se quello trovato è ottimo (in che senso?)

Terminologia e significato

Ordiniamo Allineamenti: Costo e Punteggio
Possiamo associare un costo ad una edit-string (vogliamo $\downarrow$)

Possiamo associare un punteggio ad una edit-string (vogliamo $\uparrow$)

Terminologia e significato

Ordiniamo Allineamenti: Costo e Punteggio
Possiamo associare un costo ad una edit-string (vogliamo $\downarrow$)

Possiamo associare un punteggio ad una edit-string (vogliamo $\uparrow$)

Assegnamo un Punteggio

match (buono): punteggio *alto* (e.g. 1)

substitution (cattivo): punteggio basso (e.g. 0)

insertion/deletion (cattivo): punteggio basso (e.g. 0)

Ordiniamo Allineamenti: Costo e Punteggio
Possiamo associare un costo ad una edit-string (vogliamo $\downarrow$)

Possiamo associare un punteggio ad una edit-string (vogliamo $\uparrow$)

Example
Purine (a,g) e Pirimidine (c,t) sono simili, quindi una possibile (non-banale) matrice di punteggi è la seguente:

	a	c	g	t
a	2	-1	1	-1
c	-1	2	-1	1
g	1	-1	2	-1
t	-1	1	-1	2

Ordiniamo Allineamenti: Costo e Punteggio

Possiamo associare un costo ad una edit-string (vogliamo $\downarrow$)

Possiamo associare un punteggio ad una edit-string (vogliamo $\uparrow$)

Example

BLOSUM matrice di costi:

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	−1	5																		
Asn	−2	0	6																	
Asp	−2	−2	1	6																
Cys	0	−3	−3	−3	9															
Gln	−1	1	0	0	−3	5														
Glu	−1	0	0	2	−4	2	5													
Gly	0	−2	0	−1	−3	−2	−2	6												
His	−2	0	1	−1	−3	0	0	−2	8											
Ile	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

Ordiniamo Allineamenti: Costo e Punteggio
Possiamo associare un costo ad una edit-string (vogliamo $\downarrow$)

Possiamo associare un punteggio ad una edit-string (vogliamo $\uparrow$)

Example
Penalizzazione dei gap (Affine). In effetti ci piacerebbe anche:

penalizzare l'apertura di un gap

e

penalizzare una sua estensione

... dobbiamo integrare la funzione di costo nel meccanismo ricorsivo.

Globale/Locale

La edit-string converte l'*intera* σ_2 nell'*intera* σ_1.

La edit-string converte l'*intera* σ_2 in *una porzione* di σ_1.

Distanza

Pensiamo al costo come una *distanza* tra stringhe.

Example

Distanza di Levensthein: costo 0 per **m**, 1 per gli altri.

$$\begin{array}{lccccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a & \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} & \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4. \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a & \end{array}$$

Distanza di Hamming: **i** e **d** non sono ammesse.

$$\begin{array}{lccccccccccc} \sigma_1 \equiv & a & c & g & t & c & a & t & c & a & \\ & \mathbf{s} & \mathbf{s} & \mathbf{s} & \mathbf{s} & \mathbf{s} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{m} & \Rightarrow \quad d_H(\sigma_1, \sigma_2) = 6 \\ \sigma_2 \equiv & t & a & a & g & t & g & t & c & a & \end{array}$$

La distanza di Hamming può essere usata esclusivamente su stringhe di uguale lunghezza.

Regola di massima

Che distanza devo usare?

Stringhe lunghe $\Leftrightarrow$ Levensthein

Stringhe corte $\Leftrightarrow$ Hamming

Algoritmi di Allineamento

Programmazione dinamica per l'allineamento globale

Idea!
Se dobbiamo trovare una stringa (i.e. η) determiniamone un carattere alla volta.

Programmazione dinamica per l'allineamento globale

Idea!
Se dobbiamo trovare una stringa (i.e. η) determiniamone un carattere alla volta.

- Immagino di aver determinato $\eta[1, ..., k]$ e voglio $\eta[k+1]$.

  Diciamo che $\eta[1, ..., k]$ allinei $\sigma_1[1, ..., i-1]$ e $\sigma_2[1, ..., j-1]$.
- 4 casi: $\eta[k+1] \in \{\mathbf{m}, \mathbf{s}, \mathbf{i}, \mathbf{d}\}$.

Programmazione dinamica per l'allineamento globale

Idea!
Se dobbiamo trovare una stringa (i.e. η) determiniamone un carattere alla volta.

1,2 Se $\eta[k+1] \in \{\mathbf{m}, \mathbf{s}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i]$ e $\sigma_2[1, ..., j]$.

3 Se $\eta[k+1] \in \{\mathbf{i}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i-1]$ e $\sigma_2[1, ..., j]$.

4 Se $\eta[k+1] \in \{\mathbf{d}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i]$ e $\sigma_2[1, ..., j-1]$.

Programmazione dinamica per l'allineamento globale

Idea!
Se dobbiamo trovare una stringa (i.e. η) determiniamone un carattere alla volta.

1,2 Se $\eta[k+1] \in \{\mathbf{m}, \mathbf{s}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i]$ e $\sigma_2[1, ..., j]$.

3 Se $\eta[k+1] \in \{\mathbf{i}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i-1]$ e $\sigma_2[1, ..., j]$.

4 Se $\eta[k+1] \in \{\mathbf{d}\}$, allora $\eta[1, ..., k+1]$ allinea $\sigma_1[1, ..., i]$ e $\sigma_2[1, ..., j-1]$.

Determino $\eta[k+1]$ scegliendo quello che mi da il costo minimo.

Algoritmo ricorsivo
... esponenziale

Algorithm 1: *allineamento_exp*$(\sigma_1, \sigma_2, i, j)$

if $i == 0$ *or* $j == 0$ **then** `// casi base della ricorsione`
  return $i + j$

if $\sigma_1[i] == \sigma_2[j]$ **then**
  $c_{\mathsf{m}} \leftarrow$ *allineamento_exp*$(\sigma_1, \sigma_2, i-1, j-1)$;
else
  $c_{\mathsf{s}} \leftarrow$ *allineamento_exp*$(\sigma_1, \sigma_2, i-1, j-1) + 1$;

$c_{\mathsf{i}} \leftarrow$ *allineamento_exp*$(\sigma_1, \sigma_2, i-1, j) + 1$;

$c_{\mathsf{d}} \leftarrow$ *allineamento_exp*$(\sigma_1, \sigma_2, i, j-1) + 1$;

return $\min(c_{\mathsf{m}}, c_{\mathsf{s}}, c_{\mathsf{i}}, c_{\mathsf{d}})$;

Complessità computazionale: tempo di calcolo

$$T_{allineamento\_exp}(1) = O(1)$$
$$T_{allineamento\_exp}(n) = 3\,T_{allineamento\_exp}(n-1)$$

Complessità computazionale: tempo di calcolo

$$
\begin{aligned}
T_{all\_exp}(1) &= O(1) \\
T_{all\_exp}(i+0) &= O(1) \\
T_{all\_exp}(0+j) &= O(1) \\
T_{all\_exp}(i+j) &= 2T_{all\_exp}(i+j-1) + T_{all\_exp}(i+j-2)
\end{aligned}
$$

... ricordiamo che lavoriamo nel caso *pessimo*

Complessità computazionale: spazio di memoria

Vedremo dopo. Nel frattempo osserviamo che c'è un *trade-off* tempo spazio.

(Super)Ottimizzazione

Ci sono al più $|\sigma_1| \cdot |\sigma_2|$ valori per le chiamate ricorsive ...
Posso *tenere nota* dei risultati di tutti i possibili allineamenti parziali in una matrice.

$A(i, j)$ $=$ (costo dell') allineamento di $\sigma_1[1, ..., i]$ con $\sigma_2[1, ..., j]$

Ci sono al più $|\sigma_1| \cdot |\sigma_2|$ valori per le chiamate ricorsive ...
Posso *tenere nota* dei risultati di tutti i possibili allineamenti parziali in una matrice.

$A(i,j)$ = (costo dell') allineamento di $\sigma_1[1,...,i]$ con $\sigma_2[1,...,j]$

	ϵ		j	
ϵ				
			$\vdots$	
i	$\cdots$		$d_L(i,j)$	$\cdots$
			$\vdots$	

Alogoritmo di programmazione dinamica

Algorithm 2: *allineamento_prog_din*$(\sigma_1, \sigma_2, i, j)$

for $i = 0; i \leq |\sigma_1|; i++$ **do** // inizializzo (la prima colonna di A)
  $A(i, 0) = i$

for $j = 0; j \leq |\sigma_2|; j++$ **do** // inizializzo (la prima riga di A)
  $A(0, j) = j$

// completo A procedendo $\downarrow$ sulle colonne)

for $j = 0; j \leq |\sigma_2|; j++$ **do**
  for $i = 0; i \leq |\sigma_1|; i++$ **do**
    if $\sigma_1[i] == \sigma_2[j]$ **then**
      $A(i, j) = \min\{A(i-1, j-1), A(i-1, j) + 1, A(i, j-1) + 1\};$
    else
      $A(i, j) = \min\{A(i-1, j-1) + 1, A(i-1, j) + 1, A(i, j-1) + 1\};$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\begin{array}{lcccccccccc} \sigma_1 \equiv & & a & c & g & t & c & a & t & c & a \\ & \mathbf{i} & \mathbf{m} & \mathbf{s} & \mathbf{m} & \mathbf{m} & \mathbf{s} & \mathbf{d} & \mathbf{m} & \mathbf{m} & \mathbf{m} \\ \sigma_2 \equiv & t & a & a & g & t & g & & t & c & a \end{array} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\sigma_1 \equiv \quad a \quad c \quad g \quad t \quad c \quad a \quad t \quad c \quad a$$

$$\mathbf{i} \quad \mathbf{m} \quad \mathbf{s} \quad \mathbf{m} \quad \mathbf{m} \quad \mathbf{s} \quad \mathbf{d} \quad \mathbf{m} \quad \mathbf{m} \quad \mathbf{m} \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

$$\sigma_2 \equiv \quad t \quad a \quad a \quad g \quad t \quad g \quad \quad t \quad c \quad a$$

Lettura dei risultati

- Si può leggere un allineamento sulla matrice da sx a dx e viceversa
- Si può risparmiare spazio (interessante!)
- Diverse regioni della matrice possono apparire inutili ma poi rivelarsi importanti

Perché è complesso?

Domanda:

Cosa ho *veramente* computato?

Risposta:

L'allineamento di un prefisso di σ_1 con un prefisso di σ_2

σ_1 *pattern (query)* σ_2 *testo (reference, data-base, ...)* e ci interessano i **suffissi dei prefissi di** σ_2.

Allineamento Locale

Definizione
L'allineamento locale (a distanza $\leq d$) di σ_1 in σ_2 è l'insieme di tutte le coppie $\langle i, \eta_i \rangle$ tali che η_i è l'allineamento di σ_1 e di un prefisso di $\sigma_2[i, ...]$ (a distanza $\leq d$).

Basta ...

... compilare la matrice di allineamento ponendo tutti 0 nella prima riga.

Allineamento Locale

Example

	ϵ	**a**	**c**	**g**	**t**	**a**	**c**	**g**	**c**	**g**	**t**	**a**	**c**	**a**	**g**	**t**	**a**	$\cdots$
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$
c	1	1	0	1	1	1	0	1	0	1	1	1	0	1	1	1	1	$\cdots$
g	2	2	1	0	1	2	1	0	1	0	1	2	1	1	1	2	2	$\cdots$
t	3	3	2	2	0	1	2	1	1	1	0	1	2	2	2	1	2	$\cdots$
a	4	3	3	3	1	0	1	2	2	2	1	0	1	2	3	2	1	$\cdots$

Allineamento Locale

Example

	ϵ	**a**	**c**	**g**	**t**	**a**	**c**	**g**	**c**	**g**	**t**	**a**	**c**	**a**	**g**	**t**	**a**	$\cdots$
ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$
c	1	1	0	1	1	1	0	1	0	1	1	1	0	1	1	1	1	$\cdots$
g	2	2	1	0	1	2	1	0	1	0	1	2	1	1	1	2	2	$\cdots$
t	3	3	2	2	0	1	2	1	1	1	0	1	2	2	2	1	2	$\cdots$
a	4	3	3	3	1	0	1	2	2	2	1	0	1	2	3	2	1	$\cdots$

Abbiamo introdotto un *gap inizialie a costo zero*

Rilettura e riutilizzo del risultato

La soluzione al problema dell'allineamento locale si ottiene semplicemente ragionando sul metodo impiegato per risolvere l'allineamento globale.

Funzioni di costo per i gap

- Lineare: un gap di lunghezza n ha costo pari a $\gamma(n) \equiv n \cdot g$
- Convesso: un gap di lunghezza n ha costo pari a $\gamma(n)$ tale che

$$\gamma(i+1) - \gamma(i) \leq \gamma(i) - \gamma(i-1)$$

- Affine: un gap di lunghezza n ha costo pari a $\gamma(n) \equiv g_o + (n-1) \cdot g_e$

Funzioni di costo per i gap

- Lineare: un gap di lunghezza n ha costo pari a $\gamma(n) \equiv n \cdot g$
- Convesso: un gap di lunghezza n ha costo pari a $\gamma(n)$ tale che

$$\gamma(i+1) - \gamma(i) \leq \gamma(i) - \gamma(i-1)$$

- Affine: un gap di lunghezza n ha costo pari a $\gamma(n) \equiv g_o + (n-1) \cdot g_e$

L'equazione di ricorrenza

$$A(i,j) \quad = \quad min \begin{cases} A(i-1, j-1) + cost(\sigma_1[i], \sigma_1[j]) \\ min\{A(k,j) + \gamma(i-k) \mid k = 0, \ldots, i-1\} \\ min\{A(i,k) + \gamma(j-k) \mid k = 0, \ldots, j-1\} \end{cases}$$

L'equazione di ricorrenza

Possiamo guardare indietro (nella ricorsione) quanto ci pare!

Obiettivi

Due categorie di obiettivi

- usare dati *interessanti*
- ragionare sulla ricorsione

Caratteristiche

- Il procedimento ricorsivo è inusuale
- Gli studenti tendono a "seguire" le chiamate ricorsive fino alla base ($\Rightarrow$ tabulazione)
- Una volta acquisito è uno strumento potente (torre di Hanoi)
- Consente di analizzare la complessità computazionale mediante equazioni:

$$T_{allineamento\_exp}(1) = O(1)$$
$$T_{allineamento\_exp}(n) = 3\,T_{allineamento\_exp}(n-1)$$

Tabulazione delle chiamate e spazio di memoria

Tabulazione dei risultati delle chiamate ricorsive

- Non è sempre necessaria
- Non è sempre sensato farla

La programmazione dinamica è un caso particolare: ricorsione "strutturata"

Punti importanti

- ci sono *tanti* allineamenti (tanti percorsi per trovarli) ma ...
- ... alla fine ne vogliamo uno (il migliore)!

Punti importanti

- ci sono *tanti* allineamenti (tanti percorsi per trovarli) ma ...
- ... alla fine ne vogliamo uno (il migliore)!

L'argomento si presta a:

- ragionare sull'importanza di definire chiaramente il problema: come definisco il miglior allineamento?
- ragionare come integrare conoscenza di un campo (Biologia) in uno strumento algoritmico (Computazione): come garantisco che il miglior allineamento valuti correttamente il costo dei gap?
- considerare la ricorsione come strumento per analizzare lo spazio delle soluzioni: non mi interessa come ho ottenuto la soluzione ad un sottoproblema, mi interessa solo che quel sottoproblema ha soluzione.

La tabulazione e la matrice

Analizzare la ricorsione
Tabulare i risultati e scriverli in una matrice per stimarne il numero.

La tabulazione e la matrice

Example

	ϵ	**a**	**c**	**g**	**t**	**c**	**a**	**t**	**c**	**a**
ϵ	0	1	2	3	4	5	6	7	8	9
t	1	1	2	3	3	4	5	6	7	8
a	2	1	2	3	4	4	4	5	6	7
a	3	2	2	3	4	5	4	5	6	6
g	4	3	3	2	4	5	5	5	6	7
t	5	4	4	3	2	3	4	5	6	7
g	6	5	5	4	3	3	4	5	6	7
t	7	6	6	5	4	4	4	4	5	6
c	8	7	6	6	5	4	5	5	4	5
a	9	8	7	7	6	5	4	5	5	4

$$\sigma_1 \equiv \quad a \quad c \quad g \quad t \quad c \quad a \quad t \quad c \quad a$$

$$\mathbf{i} \quad \mathbf{m} \quad \mathbf{s} \quad \mathbf{m} \quad \mathbf{m} \quad \mathbf{s} \quad \mathbf{d} \quad \mathbf{m} \quad \mathbf{m} \quad \mathbf{m} \quad \Rightarrow \quad d_L(\sigma_1, \sigma_2) = 4.$$

$$\sigma_2 \equiv \quad t \quad a \quad a \quad g \quad t \quad g \quad t \quad c \quad a$$