

Métodos de alineamiento (3)

Bioinformática, 6-3-19

Elvira Mayordomo



Hoy veremos ...

■ **Multialineamiento (MSA):**

- Un problema NP-completo
- Algoritmo aproximado para Multialineamiento:
el método de la estrella
- Métodos usados en la práctica: CLUSTAL



Alineamientos múltiples

- Hasta ahora alinéabamos (global o localmente) 2 secuencias
- Ahora, dado un conjunto de secuencias, queremos alinear **todas ellas a la vez**, de manera que los caracteres relacionados estén en la misma columna
- El problema es mucho más difícil
- A partir de 3 secuencias, se hace rápidamente inviable (por ejemplo la programación dinámica es muy lenta) → **necesitamos alternativas**



Alineamientos múltiples

- Queremos saber:
 - Cómo evaluar un MSA (cómo puntuarlo)
 - Cómo conseguir un buen MSA
- Veremos:
 - una demostración de dificultad para el problema de optimización
 - una solución aproximada
 - un ejemplo de heurística



Alineamiento múltiple

Definición: Sean $s_1 = s_{11} \dots s_{1m_1}, \dots, s_k = s_{k1} \dots s_{km_k}$ k strings sobre Σ . Sea $h: (\Sigma \cup \{-\})^* \rightarrow \Sigma^*$ la función “quita huecos”

$(h(-) = \lambda, h(a) = a \text{ para } a \in \Sigma)$

Un alineamiento múltiple de s_1, \dots, s_k es una k -tupla de strings (s'_1, \dots, s'_k) que cumplen

1. $h(s'_i) = s_i$
2. $|s'_i| = |s'_j|$ para cada i, j
3. No hay posiciones que contengan – alineados en todos los s'_i



Puntuar un MSA

- Supongamos que ya hemos conseguido un alineamiento con 3 ó más secuencias. Queremos evaluar lo bueno que es. ¿Cómo podemos calcular una puntuación?



Puntuar un MSA

- Básicamente hay dos métodos:
 1. Calculamos una secuencia consenso y la puntuación del alineamiento múltiple es la distancia a ese consenso
 2. Calculamos la puntuación del multialineamiento como la suma de las puntuaciones de los alineamientos entre parejas (SP)



Ejemplo

- Supongamos que tenemos esta alineación:

r_1 : ACGGCT

r_2 : GCGGTT

r_3 : TGGG_T

r_4 : TCGG_T

- coincidencia: +1, diferencia / indel: -1

Todas las parejas

- La matriz de puntuación:

	r_1	r_2	r_3	r_4
r_1	6	2	0	2
r_2	2	6	0	2
r_3	0	0	5	3
r_4	2	2	3	5

r_1 : ACGGCT

r_2 : GCGGTT

r_3 : TGGG_T

r_4 : TCGG_T

- Puntuación media = $(2 + 0 + 0 + 2 + 2 + 3) / 6 = 9 / 6 = 1,5$
- Nota: el alineamiento entre las secuencias r_3 y r_4 tiene una columna de “sólo gaps” – la ignoramos

coincidencia: +1, diferencia / indel: -1

secuencia de consenso

- Supongamos que representamos el alineam. por la secuencia de consenso TCGGCT
- Las puntuaciones del alineam. entre cada secuencia de entrada y el consenso:

$r_1: 4$

$r_2: 2$

$r_3: 2$

$r_4: 4$

$\text{media} = (4 + 2 + 2 + 4) / 4 = 12/4 = 3$

- Mejor permitir varias opciones o usar un modelo probabilístico como PWM

$r_1: \text{ACGGCT}$

$r_2: \text{GCGGTT}$

$r_3: \text{TGGG_T}$

$r_4: \text{TCGG_T}$

coincidencia: +1, diferencia / indel: -1



Puntuación por consenso

- Sea (s'_1, \dots, s'_k) un alineamiento múltiple de s_1, \dots, s_k con $l = |s'_i|$
- $c = c_1 \dots c_l$ es consenso de (s'_1, \dots, s'_k) si para cada j , c_j es la más frecuente de s'_{1j}, \dots, s'_{kj}
- La distancia de un alineamiento (s'_1, \dots, s'_k) a un consenso c es

$$|\{s'_{ij} \mid s'_{ij} \neq c_j\}|$$

Cuidado: buscamos distancia pequeña



Ejemplo de alineamiento múltiple

■ $s_1 = \text{AATGCT}$, $s_2 = \text{ATTC}$, $s_3 = \text{TCC}$

$s'_1 = \text{AATGCT}$

$s'_2 = \text{A-TTC-}$

$s'_3 = \text{---TCC}$



Ejemplos de puntuación

- Distancia de edición o de Levenshtein

$p(a,b)=1$ si $a \neq b$, $p(a,a)=0$, $g=1$

- A menor puntuación mejor alineamiento

Ejemplo de consenso

- $s_1 = \text{AATGCT}$, $s_2 = \text{ATTC}$, $s_3 = \text{TCC}$

$s'_1 = \text{AATGCT}$

$s'_2 = \text{A-TTC-}$

$s'_3 = \text{- - -TCC}$

$c = \text{AATTCT}$

$$\text{distancia} = (1+2+1+1+0+2)/3 = 7/3 = 2,3$$

$$p(a,b)=1 \text{ si } a \neq b, p(a,a)=0, g=1$$



Importante

- Si hay dos consensos posibles c , \underline{c} , la distancia no depende del consenso que se elija

$$\text{distancia}(c, (s'_1, \dots, s'_k)) = \text{distancia}(\underline{c}, (s'_1, \dots, s'_k))$$



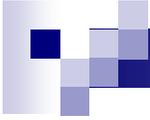
MULTCONSENSUSALIGN

- Dados s_1, \dots, s_k strings, se trata de encontrar el alineamiento múltiple que minimiza la distancia al consenso
- Hay conceptos más generales de consenso que también se usan ...



Puntuación SP

- Se trata de puntuar las parejas de alineamientos y hacer la suma
- SP = “sum of pairs”



Puntuación SP

- Sea (s'_1, \dots, s'_k) un alineamiento múltiple y δ una puntuación para alineamientos

La puntuación SP es:

$$\delta_{SP}(s'_1, \dots, s'_k) = \sum_{i=1}^k \sum_{r=i+1}^k \delta(s'_i, s'_r) / \text{no. pares}$$

Cuidado: buscamos puntuación pequeña



Ejemplo de puntuación SP

$s'_1 = \text{AATGCT}$

$s'_2 = \text{A-TTC-}$

$s'_3 = \text{---TCC}$

Recordad:

$\delta(a,a)=0$, $\delta(a,b)=1$, si $a \neq b$ (incluyendo -)

$$\delta_{\text{SP}}(s'_1, s'_2, s'_3) = (\delta(s'_1, s'_2) + \delta(s'_1, s'_3) + \delta(s'_2, s'_3)) / 3 = (3 + 5 + 3) / 3 = 11 / 3 = 3,6$$



Hoy veremos ...

- Multialineamiento (MSA):
 - **Un problema NP-completo**
 - Algoritmo aproximado para Multialineamiento:
el método de la estrella
 - Métodos usados en la práctica: CLUSTAL



MULTSPALIGN

- Dados s_1, \dots, s_k strings y δ una puntuación para alineamientos se trata de encontrar el alineamiento múltiple que minimiza la puntuación SP



Cómo calcular el óptimo SP

- Se puede utilizar la programación dinámica con matrices k dimensionales $M_{n \times n \times \dots \times n}$
- $M(i_1, \dots, i_k)$ es la puntuación del alineamiento óptimo múltiple entre
 $S_{1i_1} \dots S_{1i_1}, \dots, S_{ki_1} \dots S_{ki_1}$
- El problema es que cada posición depende de al menos 2^k vecinos
- Se tarda tiempo $O(k^2 2^k n^k)$



Cómo calcular el óptimo SP

- Si k (el número de strings a alinear) es parte de la entrada la programación dinámica es muy lenta
- De hecho el problema es NP-completo ...
- (recordad) Para NP-completos había que tomar versiones con sólo dos soluciones



DecMULTSPALIGN

- Dados s_1, \dots, s_k strings, δ una puntuación para alineamientos y d un entero positivo
- ¿Existe un alineamiento múltiple con puntuación menor o igual que d ?



NP-completitud

- Vamos a ver que DecMULTSPALIGN es NP-completo comparándolo con otro problema conocido que sabemos que es NP-completo
- Se trata del problema de la supersecuencia común



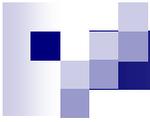
SuperSeqComun

- Dados s_1, \dots, s_k strings sobre $\{0,1\}$, m un entero
- ¿Existe un string t con $|t|=m$ de forma que s_i es subsecuencia de t para todo i ?
- Este problema es NP-completo (Middendorf 1994)

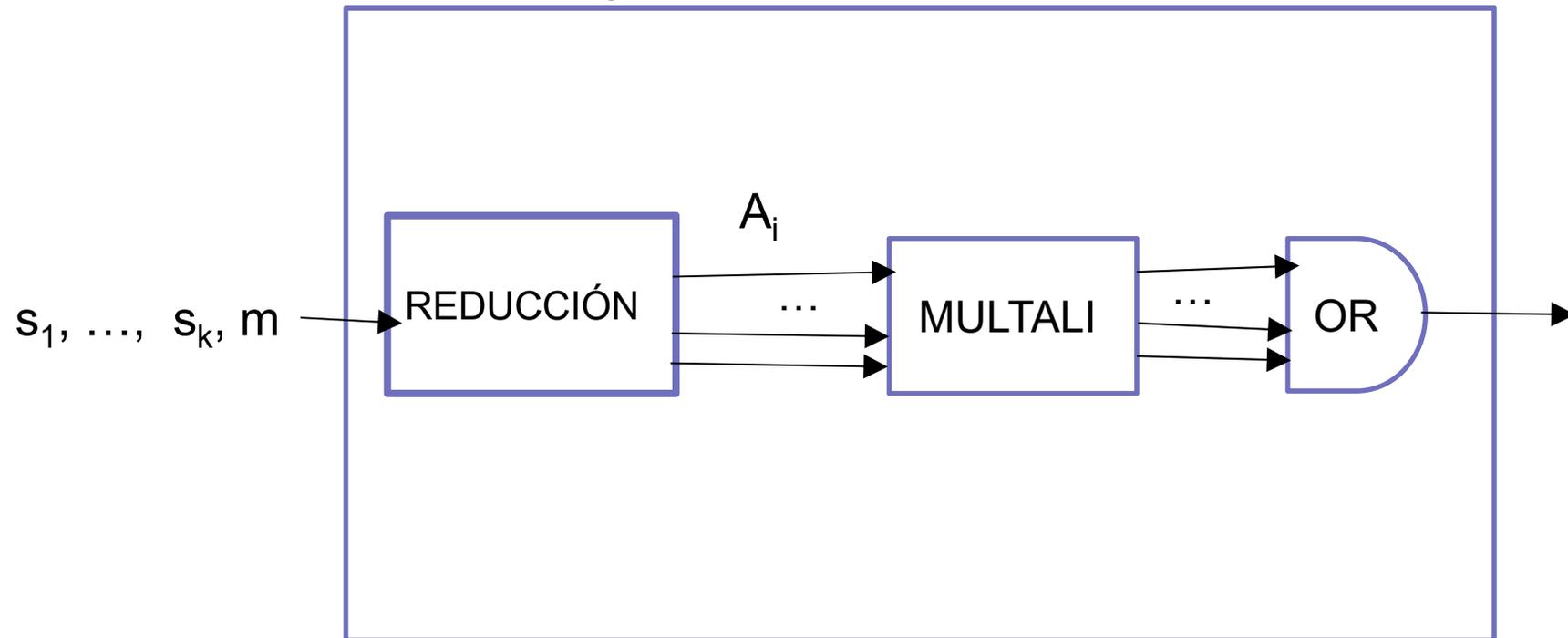


Demostración de NP-completitud

- Se trata de coger una entrada de SuperSeqComun s_1, \dots, s_k, m y convertirla en $m+1$ entradas A_0, \dots, A_m de DecMULTSPALIGN
- s_1, \dots, s_k, m tiene solución SÍ para SuperSeqComun si y sólo si **alguna** de las A_0, \dots, A_m **tiene solución SÍ** para DecMULTSPALIGN
- Así que un algoritmo eficiente para DecMULTSPALIGN nos daría un algoritmo eficiente para SuperSeqComun



Algoritmo para SuperSeqComun





Idea de la demostración

- Dados s_1, \dots, s_k strings sobre $\{0,1\}$, m un entero
- La entrada A_i será

Strings $s_1, \dots, s_k, \alpha_i, \beta_i$ con $\alpha_i = a^i, \beta_i = b^{m-i}$ (con a y b dos símbolos adicionales)

δ Puntuación que considera cerca $a,0$ y $b,1$
(ver tabla después)

$$d = (k-1)\sum |s_j| + (k+1)m$$

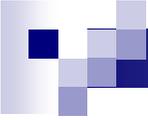


Idea de la demostración

- s_1, \dots, s_k tiene un superstring t de tamaño m

Si y sólo si para algún i

- Podemos alinear s_1, \dots, s_k , a^i , b^{m-i} con puntuación $\leq d$



Idea de la demostración

- Si tenemos un multialineamiento de $s_1, \dots, s_k, \alpha_i, \beta_i$ con puntuación $\leq d$ tenemos una supersecuencia de longitud m
- Si tenemos una supersecuencia de s_1, \dots, s_k de longitud m **que tenga i ceros**, entonces tenemos un multialineamiento de $s_1, \dots, s_k, \alpha_i, \beta_i$ con puntuación $\leq d$



Idea de la demostración

- s_1, \dots, s_k tienen un superstring t de tamaño m
Si y sólo si para algún i
- podemos alinear $s_1, \dots, s_k, a^i, b^{m-i}$ con puntuación $\leq d$
- La clave está en que δ está elegida para que si queremos conseguir la cota d
 - no podemos alinear a con b
 - no podemos alinear a con 1
 - no podemos alinear b con 0
 - la longitud del alineamiento es $\leq m$



Idea de la demostración

$$\begin{array}{c|cccc} \delta & 0 & 1 & a & b & - \\ \hline 0 & 2 & 2 & 0 & 2 & 1 \\ 1 & 2 & 2 & 2 & 0 & 1 \\ a & 0 & 2 & 0 & 3 & 1 \\ b & 2 & 0 & 3 & 0 & 1 \\ - & 1 & 1 & 1 & 1 & 0 \end{array}$$

Detalles ...



NP-completitud

- Tenemos que SuperSeqComun es NP-completo y que resolver SuperSeqComun se reduce a resolver $m+1$ entradas de DecMULTSPALIGN
- Luego DecMULTSPALIGN es NP-completo
- No podemos esperar tener algoritmos eficientes para el multialineamiento con puntuación SP ...



Hoy veremos ...

- Multialineamiento (MSA):
 - Un problema NP-completo
 - **Algoritmo aproximado para Multialineamiento: el método de la estrella**
 - Métodos usados en la práctica: CLUSTAL



¿Qué hacemos?

- No podemos esperar tener algoritmos eficientes para el multilineamiento con puntuación SP ...
- Nos conformamos con aproximar el resultado



Aproximar el alineamiento múltiple

- Se trata de combinar un conjunto de alineamientos de pares de strings
- Para ello utilizaremos el método de la estrella: ponemos como centro de la estrella el string que mejor se alinee con todos los demás y vamos extendiendo el alineamiento hasta que incluya a todas ...



MULTSPALIGN

- Dados s_1, \dots, s_k strings y δ una puntuación para alineamientos se trata de encontrar el alineamiento múltiple que minimiza la puntuación SP



Combinamos alineamientos por parejas

- El método para conseguir un multialineamiento de s_1, \dots, s_k es el *método de estrella*
- Primero seleccionamos el *centro de la estrella* c como el string más cercano a todos los demás
- Luego combinamos el alineamiento de c con cada uno de los s_i en un multialineamiento



Alineamientos compatibles

- Sea $S = \{s_1, \dots, s_k\}$ y T un subconjunto de S
- Un multi-alineamiento de S y un multi-alineamiento de T son compatibles si quitando columnas del de S obtienes el de T



Ejemplo

- $S = \{ACGG, ATG, ATCGG\}$, $T_1 = \{ACGG, ATG\}$,
 $T_2 = \{ATG, ATCGG\}$

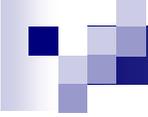
A - CGG compatible con ACGG
A - - TG A - TG
ATCGG

A - CGG no compatible con AT - G -
A - - TG ATCGG
ATCGG



Construir multi-alineamientos

- Dado un árbol de alineamientos por parejas, se pueden combinar en un multi-alineamiento
- Nosotros trataremos el caso de partir de una estrella que contiene alineamientos de cada string con el centro



Alineamiento en estrella

Algorithm 5.3 Star Alignment

Input: A set $S = \{s_1, \dots, s_k\}$ of strings.

1. Compute the center c of the star:

for $i := 1$ **to** k **do**

for $j := i$ **to** k **do**

 Compute an optimal pairwise alignment of s_i and s_j and determine its score $\text{sim}(s_i, s_j)$.

 Define c as the string t minimizing the sum $\sum_{s \in S} \text{sim}(t, s)$.

 Define T to be the star with center c and leaves $S - \{c\}$.

2. Compute a compatible multiple alignment:

for $i := 2$ **to** k **do**

 Compute a multiple alignment of c and s_1, \dots, s_i that is compatible with T from the already known compatible multiple alignment of c and s_1, \dots, s_{i-1} and the optimal pairwise alignment of c and s_i using the principle “Once a gap — always a gap”.

Output: The computed multiple alignment of S , which is compatible with T .



Once a gap --- always a gap

- Insertamos el mínimo número de – posible que permita combinar dos alineamientos de c con otras secuencias
- Veamos un ejemplo ...

Ejemplo

$c' =$ ATG-CATT
 $s'_1 =$ A-GTCAAT
 $s'_2 =$ -TCTCA--

$c'' =$ A-TGC-ATT
 $s''_3 =$ ACTGTAATT

$c''' =$ A-TG-C-ATT
 $s'''_1 =$ A--GTC-AAT
 $s'''_2 =$ --TCTC-A--
 $s'''_3 =$ ACTG-TAATT

No es óptimo, sería mejor $s'''_3 =$ ACTGT-AATT



(2-2/k)- aproximación

- Vamos a ver que el método de la estrella nos devuelve un multilineamiento que es como mucho $(2-2/k)$ veces el óptimo
- Suponemos que la puntuación δ utilizada cumple
 - $\delta(a,a)=0, \delta(-,-)=0$
 - $\delta(a,c) \leq \delta(a,b) + \delta(b,c)$
- Por ejemplo $p(a,b)=1$ si $a \neq b$, $p(a,a)=0$, $g=1$



Propiedad

- Dados $S = \{s_1, \dots, s_k\}$
sea (s'_1, \dots, s'_k) el multi-alineamiento del método estrella, sea (s''_1, \dots, s''_k) el multialineamiento óptimo, entonces

$$\delta_{SP}(s'_1, \dots, s'_k) \leq (2 - 2/k) \delta_{SP}(s''_1, \dots, s''_k)$$



Demostración

- El centro de la estrella cumple

$$\sum_i \text{sim}(c, s_i) = \min_t \sum_i \text{sim}(t, s_i)$$

donde $\text{sim}(x, y)$ es la puntuación del alineamiento óptimo del par x, y

- $M = \sum_i \text{sim}(c, s_i)$
- Suponemos que $c = s_k$



Demostración

$$\delta_{SP}(s'_1, \dots, s'_k) = \sum_{i=1}^k \sum_{r=i+1}^k \delta(s'_i, s'_r)$$

$$2 \delta_{SP}(s'_1, \dots, s'_k) = \sum_{i=1}^k \sum_{r=1}^k \delta(s'_i, s'_r) \leq \sum_{i=1}^k \sum_{r=1}^k \delta(s'_i, c') + \delta(c', s'_r) =$$

(ya que (s'_i, c') es óptimo para (s_i, c))

$$\sum_{i=1}^{k-1} \sum_{r=1}^{k-1} \text{sim}(s_i, c) + \text{sim}(c, s_r) =$$
$$2(k-1) \sum_{i=1}^k \text{sim}(s_i, c) = 2(k-1) M$$



Demostración

$$\delta_{SP}(s''_1, \dots, s''_k) = \sum_{i=1}^k \sum_{r=i+1}^k \delta(s''_i, s''_r)$$

$$2 \delta_{SP}(s''_1, \dots, s''_k) = \sum_{i=1}^k \sum_{r=1}^k \delta(s''_i, s''_r) \geq$$

$$\sum_{i=1}^k \sum_{r=1}^k \text{sim}(s_i, s_r) \geq$$

$$k \sum_{i=1}^k \text{sim}(s_i, c) = k M$$



Demostración

$$2 \delta_{SP}(s'_1, \dots, s'_k) \leq 2(k-1) M$$

$$2 \delta_{SP}(s''_1, \dots, s''_k) \geq k M$$

Luego

$$\delta_{SP}(s'_1, \dots, s'_k) \leq (2-2/k) \delta_{SP}(s''_1, \dots, s''_k)$$



Conclusión

- El método de la estrella da un multialineamiento de coste de algo menos que el doble del óptimo
- También se puede usar el mismo método con puntuación por consenso



Hoy veremos ...

- Multialineamiento (MSA):
 - Un problema NP-completo
 - Algoritmo aproximado para Multialineamiento: el método de la estrella
 - **Métodos usados en la práctica: CLUSTAL**



MSA en la práctica: heurísticas

- Muchos métodos:

- Clustal (ClustalW, ClustalX, Clustal Omega, etc.)
- T-Coffee
- MAFFT
- MUSCLE
- ...

- Vamos a estudiar las ideas principales de Clustal



Clustal

- Primero se propuso por Giggins y Sharp en 1988
- La versión popular ClustalW (Clustal **w**eighted) fue propuesta por Thompson et al en 1994
- Pasos principales:
 - Calcular la matriz de las distancias entre cada par de secuencias
 - Construir un árbol que captura la relación entre las secuencias según la matriz de distancia
 - Progresivamente alinear las secuencias según el árbol

Matriz de distancias

- matriz de distancias: una distancia entre cada par de secuencias
- Digamos que utilizamos programación dinámica para conseguir un alineam. óptimo y la distancia = longitud del alineam. - puntuación del alineam.
 - no tenemos todavía el MSA ...

r_1 : ACGGCT

r_2 : GCGGTT

r_3 : TGGGT

r_4 : TCGGT

	r_1	r_2	r_3	r_4
r_1	0	4	6	4
r_2	4	0	6	4
r_3	6	6	0	2
r_4	4	4	2	0

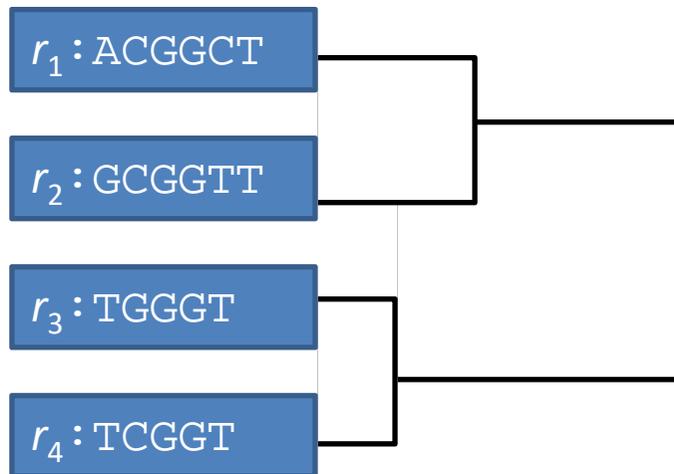


De matriz de distancia a árbol

- **Árbol:** las secuencias cercanas se colocan cerca en el árbol, la distancia en el árbol se corresponde a la distancia de la matriz
- **Definir un árbol (una forma posible):** agrupar repetidamente las dos secuencias más cercanas entre sí (clustering jerárquico)
 - Veremos más sobre filogenética ...

De matriz de distancia a árbol

- Un árbol posible:



r_1 : ACGGCT

r_2 : GCGGTT

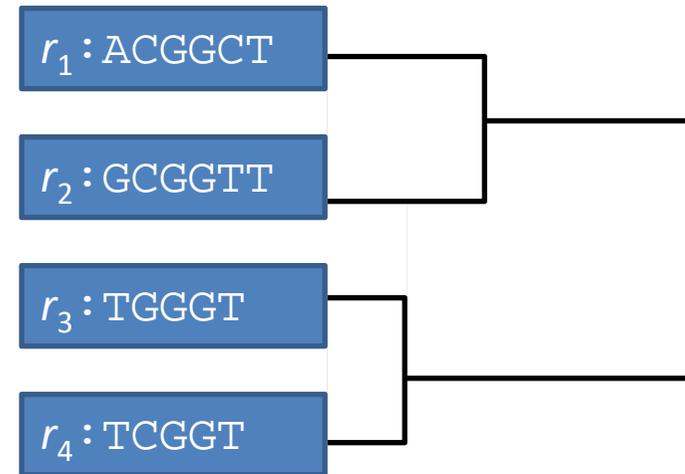
r_3 : TGGGT

r_4 : TCGGT

	r_1	r_2	r_3	r_4
r_1	0	4	6	4
r_2	4	0	6	4
r_3	6	6	0	2
r_4	4	4	2	0

De árbol a MSA

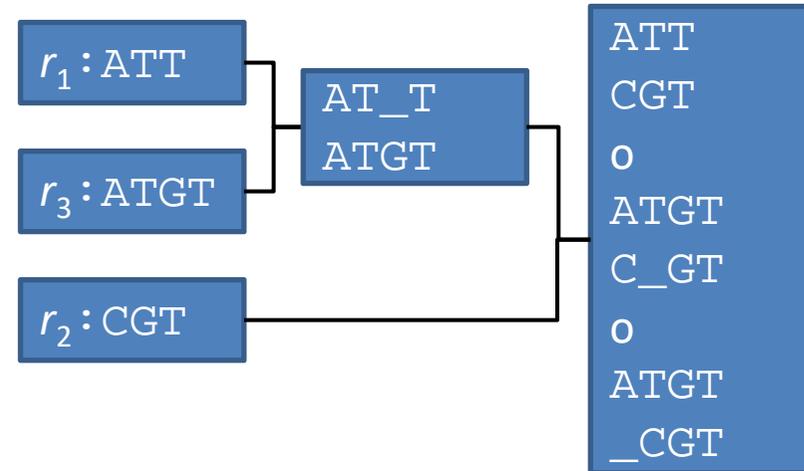
- Orden del alineam.:
 1. r_3 vs. r_4
 2. r_1 vs. r_2
 3. (r_1, r_2) vs. (r_3, r_4)
 - Puedo alinear la sec. de consenso entre r_1 y r_2 con la sec. de consenso entre r_3 y r_4
 - (hay alternativas)



Un ejemplo completo

- $r_1 = \text{ATT}$, $r_2 = \text{CGT}$, $r_3 = \text{ATGT}$
- Coincid.: 1; difer.: -1; indel: -2
- Mejor alineam. entre r_1 y r_2 :
 ATT
 CGT
 (puntuac. = -1, distancia = 4)
- Mejor alineam. entre r_1 y r_3 :
 AT_T
 ATGT
 (puntuac. = 1, distancia = 3)
- Mejores alineams. entre r_2 y r_3 :
 C_GT $_ \text{CGT}$
 ATGT y ATGT
 (puntuac. = -1, distancia = 5)
- Consenso entre r_1 y r_3 :
 $r_{13} = \text{ATT}$ o ATGT
- Mejores alineamientos entre r_{13} y r_2 :
 ATT ATGT ATGT
 CGT ó C_GT ó $_ \text{CGT}$
 (para ATT) (para ATGT)

Árbol resultante:



MSAs:

AT_T	AT_T	AT_T
CG_T	C_GT	_CGT
ATGT ó	ATGT ó	ATGT



Posible trabajo

- Investigar MAFFT



Resumen

- Necesitamos métodos heurísticos de alineam. debido a que la programación dinámica es inviable para secuencias muy largas y / o muchas secuencias
- Para alineamiento de pares, FASTA y BLAST primero encuentran alineam. locales, a continuación, los extienden / combinan
 - Se puede evaluar la significancia estadística de los resultados
- Para MSA, una heurística es llevar a cabo una serie de alineamientos por pares, de forma voraz