Biologia Computacional - 2004/2 11/23/04

Aula 3: Alinhamento Local

Instrutor: Berilhes Borges Garcia Escriba: Suelen Marconsini Loureiro

1 Introdução

Alinhamento global é muito utilizado para comparar membros de uma mesma família de proteínas:

- frequentemente de tamanho similar (ex: globulinas).
- de modo a inferir a história evolutiva.

Por outro lado, alinhamento local é mais útil para comparar sequências de DNA de espécies diferentes e comparar proteínas de famílias diferentes.

O problema do alinhamento local com relação a duas strings $S_1[1...n]$ e $S_2[1...m]$ pode ser resolvido em tempo $O(nm)$ (Smith & Waterman 1981).
Note que existem $\theta(n^2m^2)$ pares possíveis de substrings!

Suposição útil: a similaridade de duas strings vazias é zero.

2 Alinhamento local de sufixos

Considere primeiro uma versão mais restrita do problema:

Dado índices $i \leq n$ e $j \leq m$, o problema do alinhamento local de sufixos consiste em encontrar um sufixo α de $S_1[1...i]$ e um sufixo β de $S_2[1...j]$ de similaridade máxima (que nós denotamos por v(i,j)).

Exemplo: Assuma que os escores são $s(x, y) = 2$ quando $x = y$, e $s(x, y) = -1$ quando $x \neq y$ (para qualquer $x, y \in \sum'$).

Considere as strings:

	1	2	3	4	5	6	7
S_1	a	b	c	f	d	e	f
S_2	f	f	f	c	d	e	

Então:

- v(3,4) = 2 ($\alpha = \beta = c$)
- v(4,5) = 1 ($\alpha = cf$, $\beta = cd$)

- v(5,5) = 3 ($\alpha^{'} = f\_d$, $\beta = fcd$)

Denote o alinhamento local ótimo por v^*.

Teorema 1.

$$v^* = max\{v(i,j)|i \leq\ n, j \leq\ m\} \tag{1}$$

Proof. 1. $v^* \geq\ max\{v(i,j)|i \leq\ n, j \leq\ m\}$ (alinhamento local de sufixos é um caso especial do alinhamento local).

2. Assuma que v^* é a similaridade das substrings α e β terminando nas posições i^* e j^*,respectivamente. Isto implica que α e β são sufixos de $S_1[1...i^*]$ e $S_2[1...j^*]$, respectivamente. Portanto, $v^* \leq\ max\{v(i,j)|i \leq\ n, j \leq\ m\}$.

3. (1) e (2) implicam o teorema.

□

Corolário 1. *Se $v(i^*, j^*) = max\{v(i,j)|i \leq\ n, j \leq\ m\}$, então sufixos α de $S_1[1...i^*]$ e β de $S_2[1...j^*]$, cuja similaridade é $v(i^*, j^*)$, formam uma solução para o problema do alinhamento local.*

- Como determinar i^*,j^*,α e β?

  Por programação dinâmica.

2.1 Recorrências para o alinhamento local de sufixos

Caso base: $v(i,0) = v(0,j) = 0$ (você sempre pode escolher a string vazia)

Caso indutivo: $(i,j > 0)$

Como os sufixos α de $S_1[1...i]$ e β de $S_2[1...j]$ podem ser alinhados de forma ótima? Existem 4 possibilidades.

1. Eles poderiam ser strings vazias → escore $v(i,j) = 0$
2. $S_1[i]$ contra $S_2[j]$ → escore $v(i-1, j-1) + s(S_1[i], S_2[j])$
3. $S_1[i]$ contra espaço → escore $v(i-1, j) + s(S_1[i], \_)$
4. $S_2[j]$ contra espaço → escore $v(i, j-1) + s(\_, S_2[j])$

O ótimo para $v(i,j)$ é obtido escolhendo-se o máximo das possibilidades acima.

$$v(i,j) = max \begin{cases} 0 \\ v(i-1, j-1) + s(S_1[i], S_2[j]) \\ v(i-1, j) + s(S_1[i], \_) \\ v(i, j-1) + s(\_, S_2[j]) \end{cases} \tag{2}$$

A tabela de $v(i,j)$, com ponteiros, pode ser construída utilizando-se esra recorrência, exatamente como antes.

O valor máximo v^* é encontrado procurando todas as células da tabela, digamos que seja a célula (i^*, j^*). Substrings α e β com similaridade v^* são encontradas seguindo-se os ponteiros da célula (i^*, j^*) para uma célula $(i^{'}, j^{'})$ com $v(i^{'}, j^{'}) = 0$. Então $\alpha = S_1[i^{'}...i^*]$ e $\beta = S_2[j^{'}...j^*]$.

Teorema 2. *O problema do alinhamento local entre as strings $S_1[1...n]$ e $S_2[1...m]$ pode ser resolvido em tempo $O(nm)$.*

Observações:

- Ao invés de um par (α, β) de substrings, um certo número de substrings similares, digamos com similaridade acima de um certo valor, podem ser encontrados de forma semelhante.
- Note que esquemas de pontuação apropriados são necessários para encontrar alinhamentos locais significativos.

3 Gaps

Um gap é uma sequência consecutiva de espaços em um alinhamento.
Alinhamentos com gaps correspondem melhor a alguns fenômenos biológicos que nós tentamos modelar.

- Uma deleção ou inserção de uma substring de DNA "frequentemente"ocorre como um evento mutacional único.
- Gaps podem algumas vezes representar características importantes para inferir a história evolutiva de um conjunto de strings.

Como penalizar os gaps?
Existem várias possibilidades de atribuição de custos aos gaps em um alinhamento: constante, afim, convexo e arbitrário.

O modelo constante é o mais simples.

Ajuste $s(\_, x) = s(x, \_) = 0$ para todo caracter x, e a cada gap atribua um peso W_g (independente do tamanho do gap).
Assim nós temos que encontrar um alinhamento que maximize

$$\sum_{i=1}^{l} (s(S_1^{'}[i], S_2^{'}[i])) - G.W_g$$

onde G é o número total de gaps.