Algoritmos para Análise de Sequências Biológicas

Alinhamento múltiplo de sequências

Sumário

- como calcular o score de um alinhamento múltiplo
- estratégias para alinhamento múltiplo

Avaliação de soluções

Heurística SP (soma de pares) soma do score das substituições para cada par de sequências

Consenso calcular nº de caracteres em cada posição que coincidem com o consenso

Algoritmos de alinhamento múltiplo

Generalização de programação dinâmica só pode ser utilizado para um nº limitado de sequências de tamanho reduzido

Progressivos iniciam com 2 sequências e vão adicionando as restantes

Iterativos consideram um alinhamento inicial que vai sendo melhorado

Híbridos podem usar outros tipos de informação (e.g. estrutura 3D)

Programação dinâmica

- Matriz passa a ser N dimensional
- Complexidade é proporcional ao produto do tamanho das sequências
- Só pode ser utilizado para **poucas** sequências de tamanho **pequeno**

PD para AMs

```
S(i,j,k) = max(
    S(i - 1, j - 1, k - 1) + s(a[i], b[j], c[k]),
    S(i - 1, j - 1, k    ) + s(a[i], b[j],   _ ),
    S(i - 1, j    , k - 1) + s(a[i],   _ , c[k]),
    S(i    , j - 1, k - 1) + s(  _ , b[j], c[k]),
    S(i - 1, j    , k    ) + s(a[i],   _ ,   _ ),
    S(i    , j - 1, k    ) + s(  _ , b[j],   _ ),
    S(i    , j    , k - 1) + s(  _ ,   _ , c[k]))
```

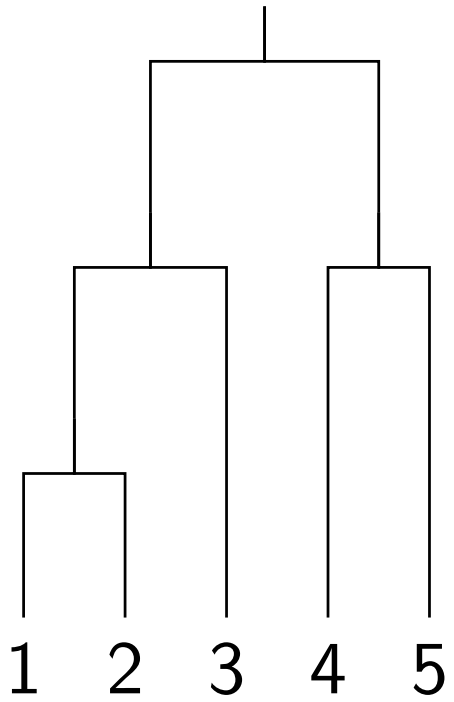
Algoritmos Progressivos

- Inicialmente, calculam-se os alinhamentos para todos os pares de sequências possíveis
- Constrói-se uma árvore filogenética de acordo com o mérito (ou distância) dos alinhamentos anteriores
- Escolhem-se as duas sequências mais similares e faz-se o seu alinhamento
- Seguindo a estrutura da árvore vão-se adicionando as restantes sequências ao AM

Exemplo de uma matriz de distâncias

	$\{s_1\}$	$\{s_2\}$	$\{s_3\}$	$\{s_4\}$	$\{s_5\}$
$\{s_1\}$	0				
$\{s_2\}$	2	0			
$\{s_3\}$	5	4	0		
$\{s_4\}$	7	6	4	0	
$\{s_5\}$	9	7	6	3	0

Exemplo da árvore filogenética



1. Alinha-se s_1 com s_2
2. Adiciona-se s_3 ao alinhamento entre s_1 e s_2
3. Alinha-se s_4 com s_5
4. Adiciona-se os dois alinhamentos parciais

Combinação de alinhamentos

Várias alternativas para realizar este processo:

- Calcular consensos dos alinhamentos e alinhar estes consensos (por ex. usando PD)
- Adaptar a PD para trabalhar com matrizes de ocorrência/ perfis
- Ter uma sequência base em todos os AMs e basear a combinação nas posições dessa sequência
- Espaçamentos mantêm-se: once a gap, always a gap

Problemas com o AM progressivo

- Forte dependência da ordem das sequências, especialmente das iniciais
  - Corre bem se estas forem muito similares
  - Erros propagam-se para o AM caso contrário
- Difícil escolher matrizes de substituição e penalizações para espaçamentos – solução: parâmetros variam dinamicamente: -Modelo de penalização de espaçamentos varia conforme a posição (aminoácidos nas redondezas; gaps já existentes)
- Matrizes de substituição variam conforme a divergência das sequências
- Estes algoritmos têm problemas com sequências mais distantes.

Implementação do AM progressivo em Python

- Vamos usar a ordem pré-definida das sequências
- Usamos o algoritmo Needleman-Wunsch para o alinhamento
- Alinhamos as 2 primeiras sequências usando NW
- Calculamos a sequência de consenso
- Adicionamos uma sequência de cada vez pela ordem dada
- Calculamos a sequência de consenso de cada vez

Exemplo

Sequências

```
s1: ATAGC
s2: AACC
s3: ATGAC
```

Alinhar s_1 e s_2

```
s1:   ATAGC
s2:   A-ACC

cons: ATAGC
```

Adicionar s_3

```
AM:   ATAG-C
s3:   AT-GAC

cons: ATAGAC
```

Alinhamento múltiplo

```
s1: ATAG-C
s2: A-AC-C
s3: AT-GAC
```