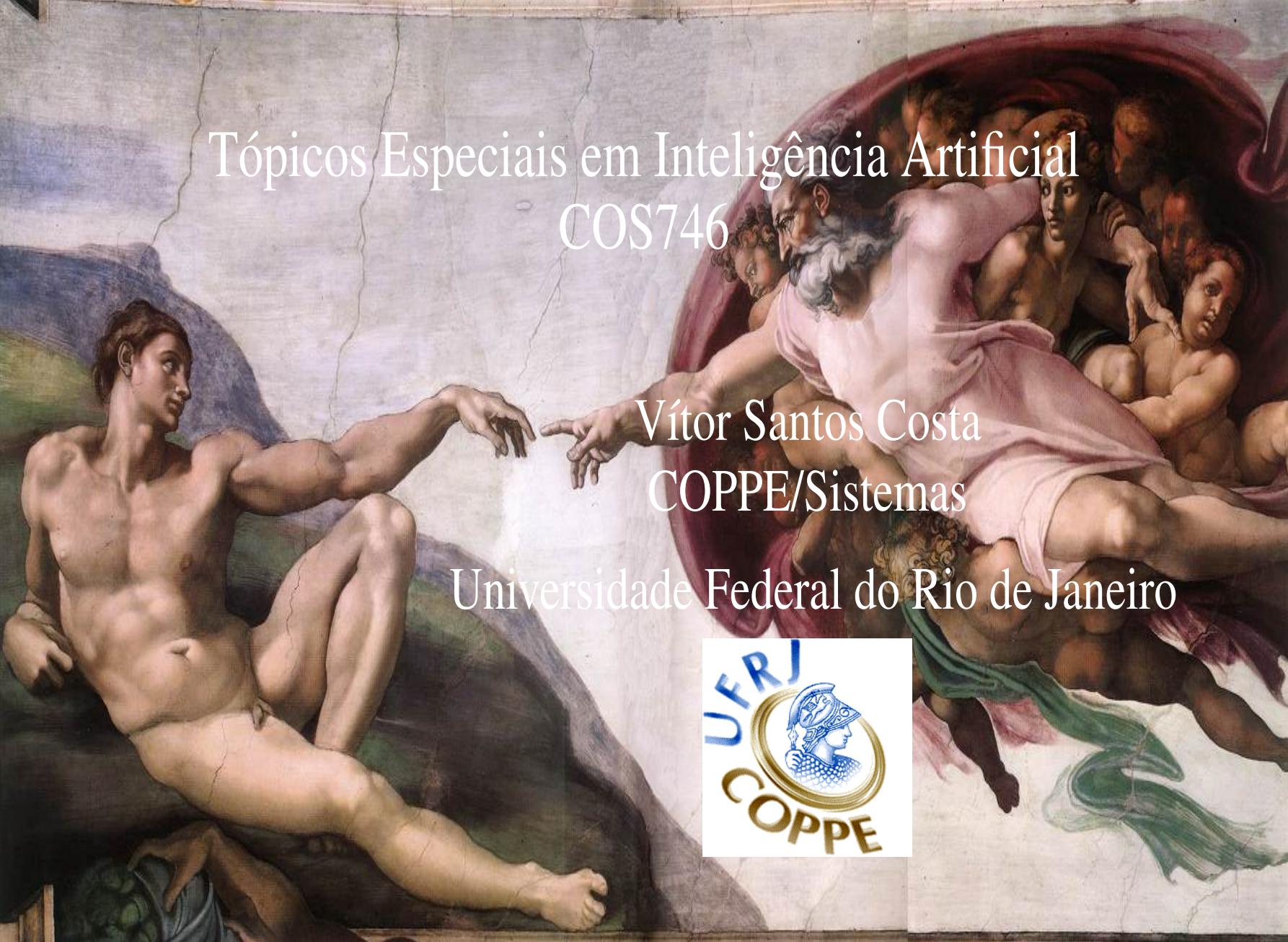


Tópicos Especiais em Inteligência Artificial
COS746

Vítor Santos Costa
COPPE/Sistemas
Universidade Federal do Rio de Janeiro

Agradecimento

- Copiado dos slides de Mark Craven para BMI/CS 576, UW-Madison

Alinhamento de Pares de Sequências

Dada:

- Um par de sequências (DNA ou proteína)
- um método para calcular a similaridade de um par de caracteres na sequência

Faça

- Encontre as correspondências entre subsequências nas sequências que maximizam uma *função de semelhança*.

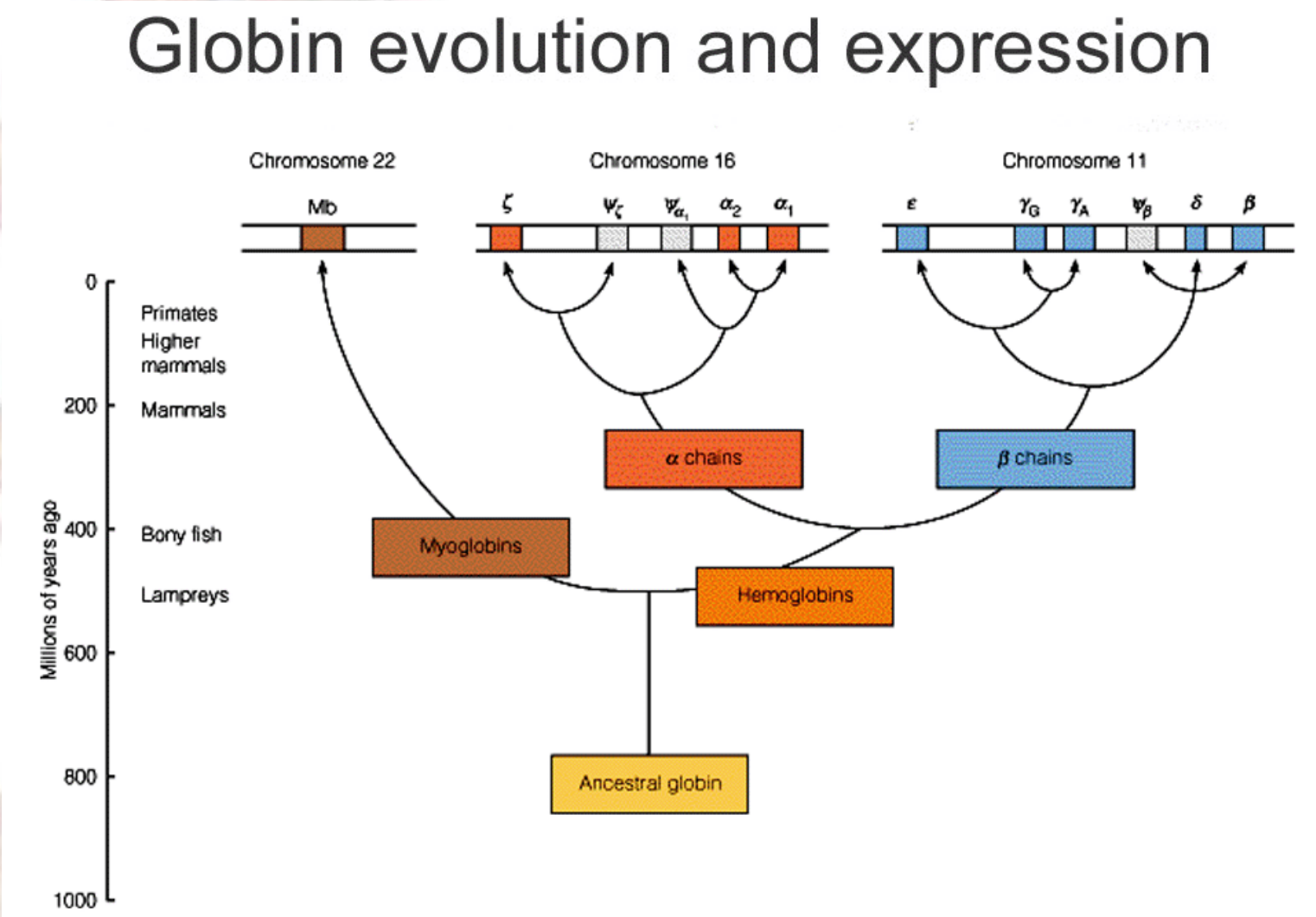
Motivação

- Comparar sequências para obter informação sobre a estrutura e função de uma sequência.
- Juntar um conjunto de fragmentos de sequência
- Comparar um segmento sequenciado por diferentes laboratórios

A Importância de Homologia

- *Homologia*: semelhança causada por descendência do mesmo antepassado
- Muitas vezes podemos inferir homologia de similaridade
- Utilidade: inferir estrutura/função a partir de similaridade.

Exemplo: Globinas



Homologia

- Sequências homologas podem ser divididas em dois grupos:
  - ⋆ Ortólogas: divergiram para espécies diferentes (eg, α-globina humana e do rato)
  - ⋆ Parálogas: divergiram devido a duplicação de genes na mesma espécie (eg, as várias versões da α-globina humana e da β-globina humana).

Problemas no Alinhamento de Sequências

- As sequências que vamos comparar provavelmente diferem em tamanho
- Pode haver apenas uma pequena região nas sequências que alinha
- queremos permitir alinhamentos parciais: por ex, alguns pares de amino-ácidos são mais substituíveis do que outros
- regiões de tamanho variável podem ter sido inseridas ou removidas do antepassado comum.

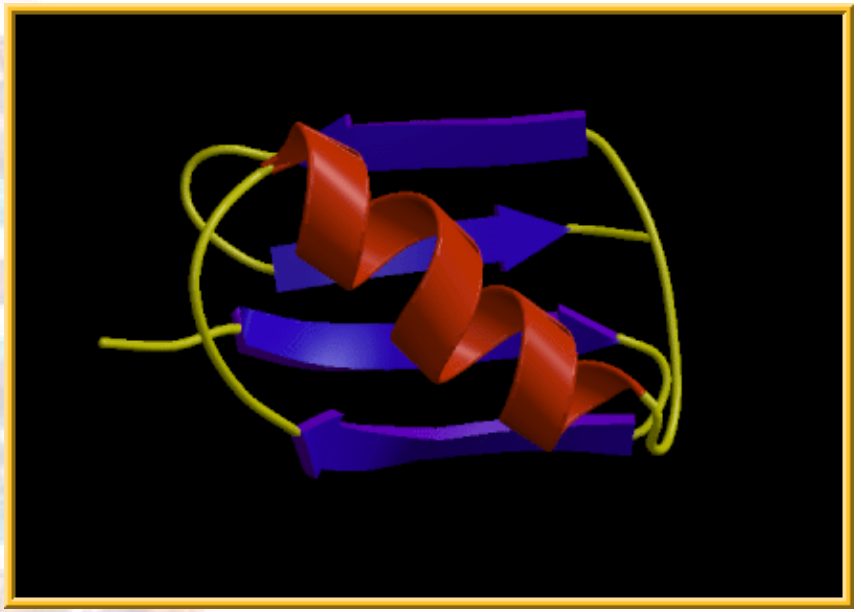
Buracos

- Sequências podem ter divergido do antepassado comum através de vários tipos de mutações:
  - ⋆ Substituições: $(ACGA \longrightarrow AGGA)$
  - ⋆ Inserções: $(ACGA \longrightarrow ACCGA)$
  - ⋆ Remoções: $(ACGA \longrightarrow AGA)$
- os últimos dois casos correspondem a buracos no alinhamento.

Inserções e Remoções vs Estrutura da Proteína

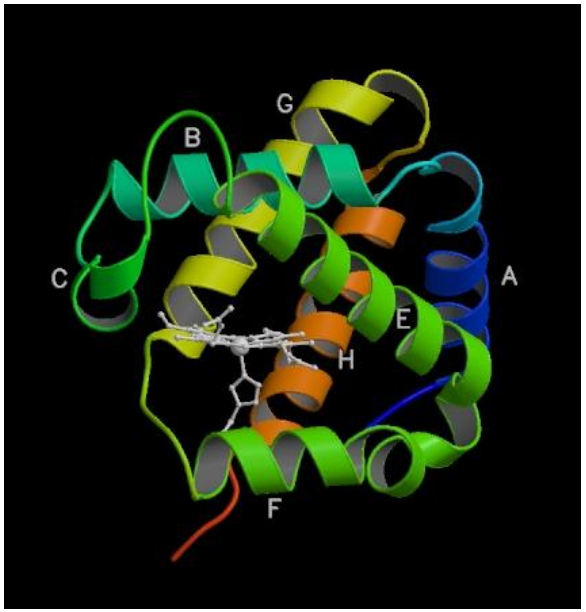
Porque é que duas sequências semelhantes podem ter muitas inserções ou remoções

- Inserções e remoções podem não afectar significativamente a estrutura da proteína.



Exemplo de Alinhamento: Globinas

- À direita, estrutura tipo de Globinas
- Em baixo, parte do Alinhamento para oito globinas




```
                A0   A4   A8   A12            B1        B6        B14  C2        CD1  CD4
                ↓    ↓    ↓    ↓              ↓         ↓         ↓    ↓         ↓    ↓
Hb_a  ----------VLSPADKTNVKAAWGKVGA----HAGEYGAEALERMFLSFPTTKTYFPHF
Hb_b  ----------VHLTPEEKSAVTALWGKV------NVDEVGGEALGRLLVVYPWTQRFFESF
Mb_SW ----------VLSEGEWQLVLHVWAKVEA----DVAGHGQDILIRLFKSHPETLEKFDRF
LegHb ---------GALTESQAALVKSSWEEFNA----NIPKHTHRFFILVLEIAPAAKDLFSFL
BacHb --------LDQQTINIIKATVPVLKEHG------V-TITTTFYKNLFAKHPEVRPLF---
SeaHb GGTLAIQAQGDLTLAQKKIVRKTWHQLMR----NKTSFVTDVFIRIFAYDPSAQNKFPQM
AscHb ------------ANKTRELCMKSLEHAKVDTSNEARQDGIDLYKHMFENYPPLRKYFKS-
Eryt. ----------LSADQISTVQASFDKVKG---------DPVGILYAVFKADPSIMAKFTQF
```

Tipos de Alinhamento

- *Global*: encontrar o melhor alinhamento entre sequências completas
- *Local*: encontrar o melhor alinhamento entre subsequências completas
- *Semi-Global*: encontrar o melhor alinhamento sem penalizar espaços brancos nas bordas do alinhamento

Como Avaliar um Alinhamento

- Matriz de substituição:
  - $\star$ $s(a, b)$ indica o preço de alinhar o caracter a com o caracter b.
- Função de penalização de intervalos:
  - $\star$ $w(k)$ indica o custo de um intervalo de tamanho k.

Função de Penalização Linear

- Diferentes funções de penalização podem requerer algoritmos de programação dinâmica diferente
  - ⋆ O caso mais simples é quando usamos uma função linear:

$$w(k) = gk$$

onde g é uma constante

- Vamos começar por aqui.

Pontuação de Alinhamentos

- A pontuação de um alinhamento é:
  1. somatório dos pares de caracteres alinhados,
  2. mais pontuação para buracos
- Exemplo: dado o alinhamento

```
VAHV---D--DMPNALSALSDLHAHKL

AIQLQVTGVVVTDATLKNLGSVHVSKG
```

- a pontuação será:

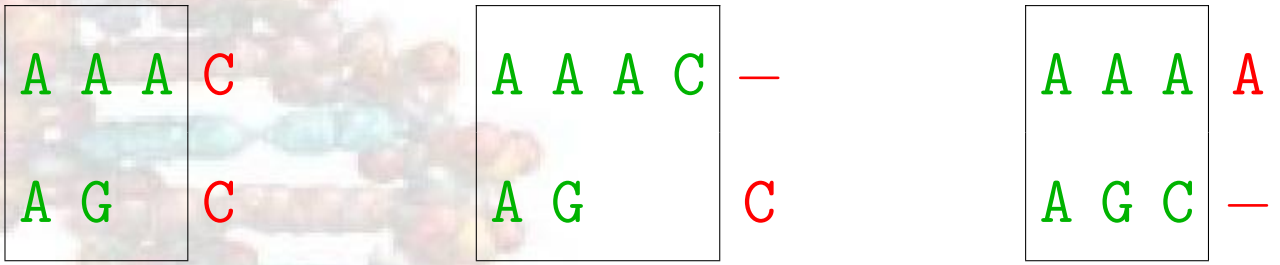
$$s(\mathtt{V},\mathtt{A}) + s(\mathtt{A},\mathtt{I}) + s(\mathtt{H},\mathtt{Q}) + 3g + s(\mathtt{D},\mathtt{G}) + \ldots$$

Alinhamento de Pares por Programação Dinâmica

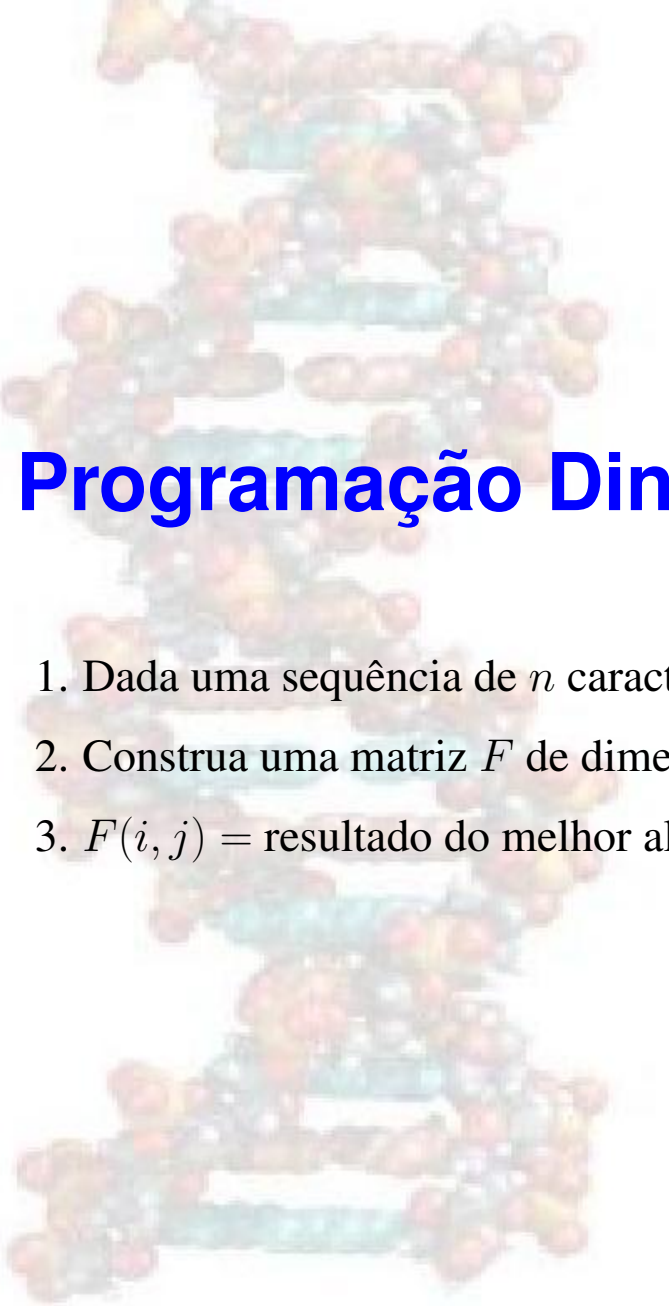
- Needleman & Wunsch, *Journal of Molecular Biology*, 1970
- *Programação Dinâmica*: resolver uma instância de um problema usando soluções computadas para pequenas partes do problema.
- Ideia: determinar alinhamento óptimo de duas sequências determinando o melhor alinhamento para todos os prefixos.

Alinhamento de Pares por Programação Dinâmica

- Considere o último passo na computação do alinhamento de `AAAC` com `AGC`
- Três opções possíveis:



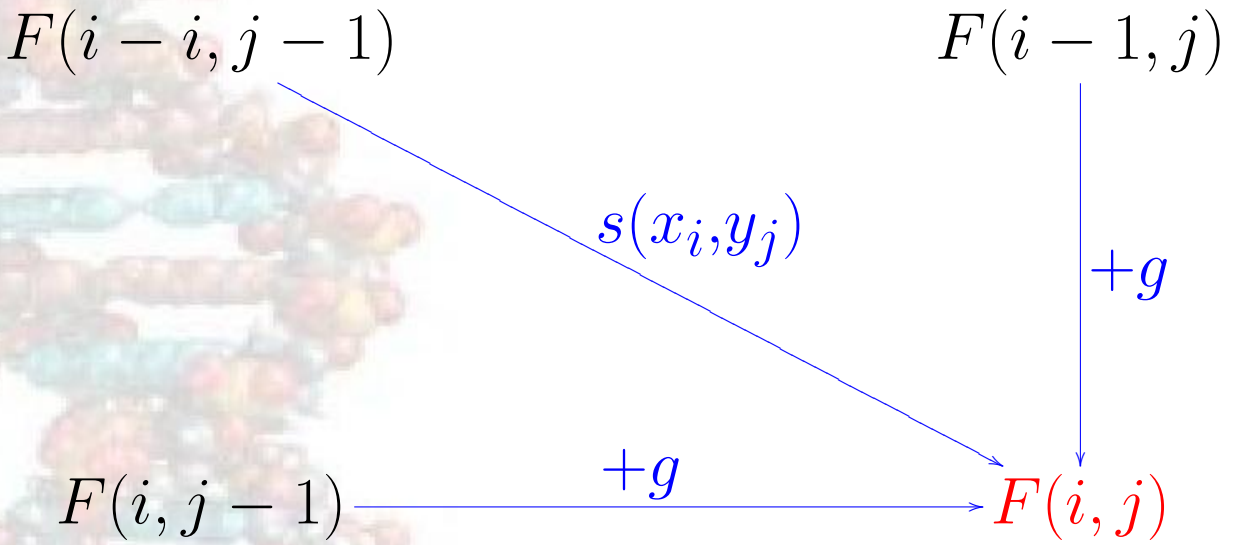
- Considere:
  1. Melhor Alinhamento dos Prefixos +
  2. Resultado do Alinhamento do par



Programação Dinâmica

1. Dada uma sequência de n caracteres x e uma sequência de m caracteres y,
2. Construa uma matriz F de dimensão $(n+1) \times (m+1)$
3. $F(i,j) =$ resultado do melhor alinhamento de $x[1 \ldots i]$ com $y[1 \ldots j]$.

Programação Dinâmica: Ideia Básica

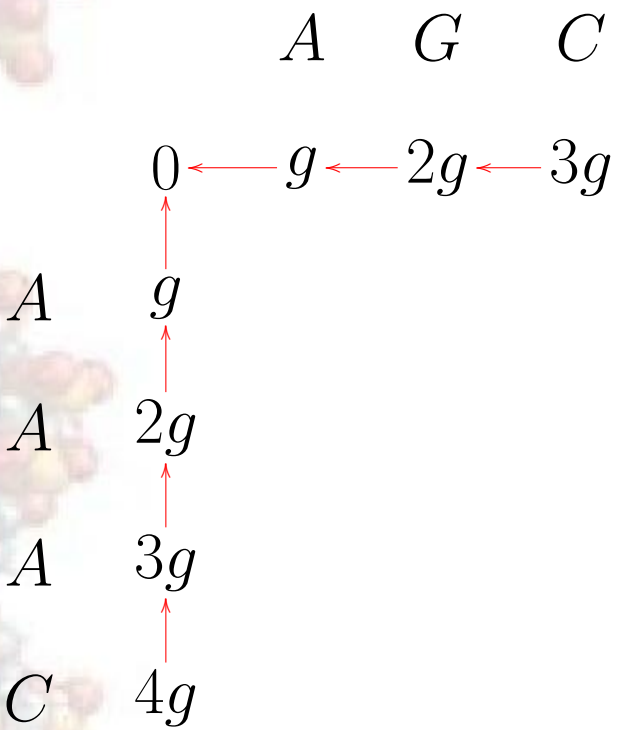


Algoritmo para Alinhamento Global com Penalização Linear de Buracos

Uma maneira é especificar a DP através da sua relação de recorrência:

$$F(i,j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) + d \\ F(i, j-1) + d \end{cases}$$

Inicialização da Matriz



Esquema do Algoritmo

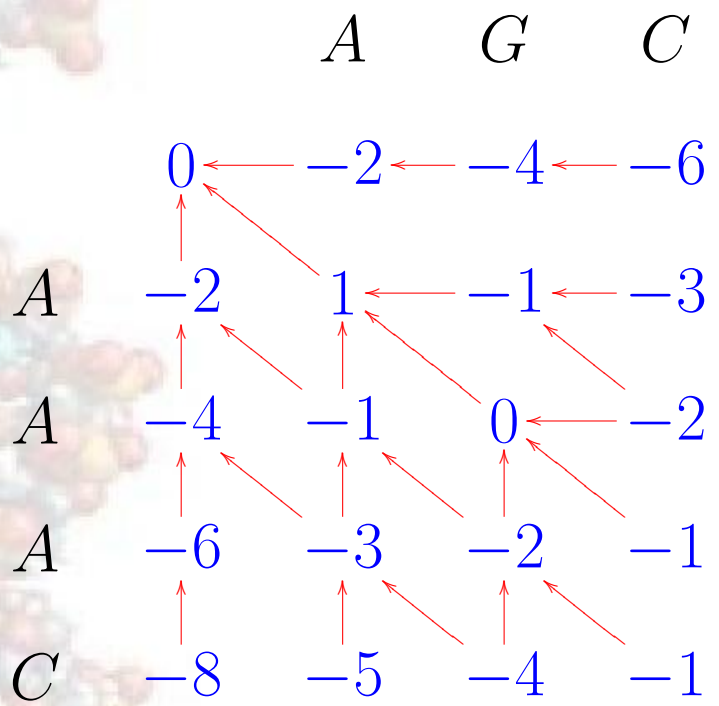
- inicializar primeira linha e coluna da matriz
- preencher o resto da matriz de cima para baixo, e esquerda para a direita
- para cada $F(i, j)$, guarde ponteiro para célula que deu o melhor resultado
- $F(m, n)$ tem a pontuação de alinhamento óptima: siga os ponteiros desde $F(m, n)$ até $F(0, 0)$ para recuperar o alinhamento.

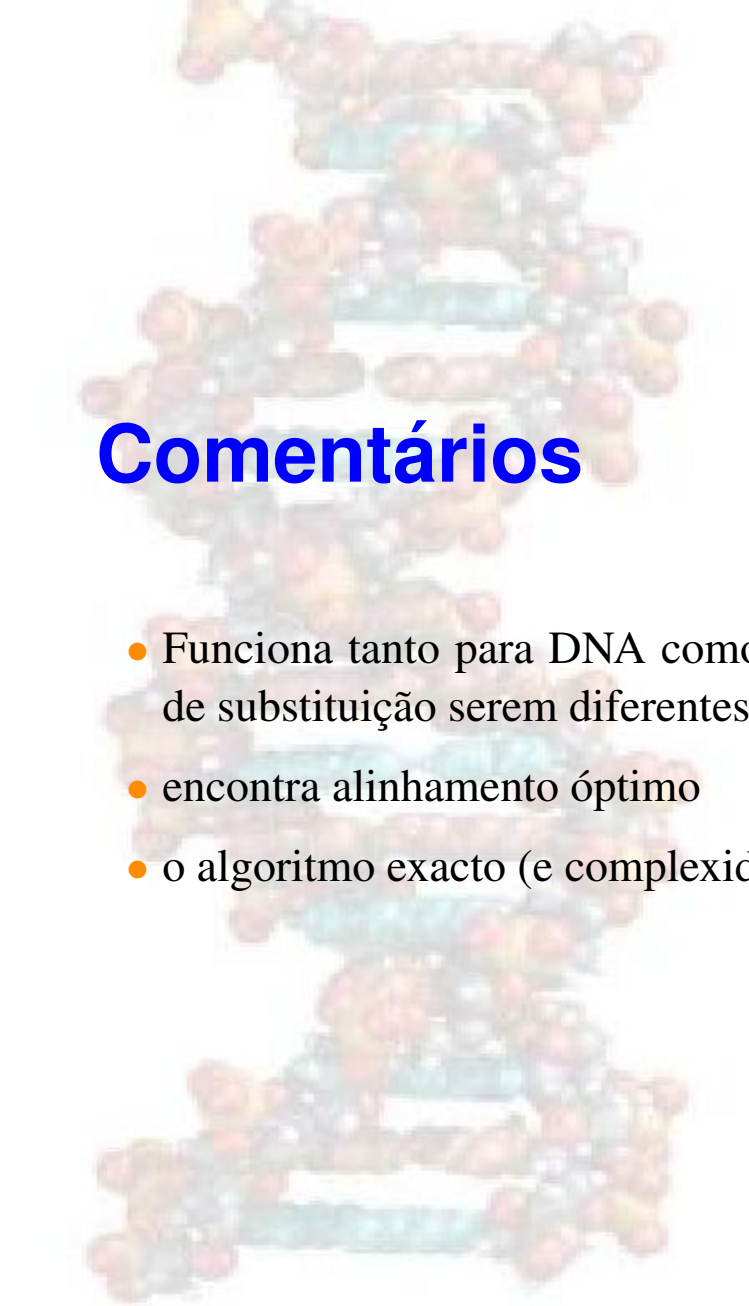
Exemplo do Esquema do Algoritmo

Imagine que escolhíamos o seguinte esquema de pontuação:

- acerto: $+1$
- erro: -1
- $g(\text{penalidade para alinhar com um buraco}) = -2$

Exemplo do Esquema do Algoritmo



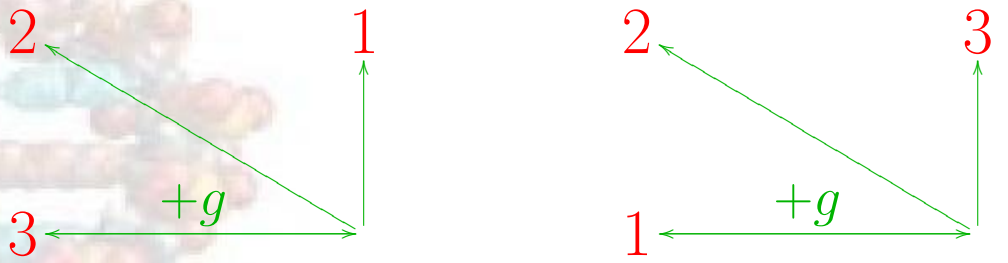


Comentários

- Funciona tanto para DNA como para sequências de matrizes, apesar das matrizes de substituição serem diferentes
- encontra alinhamento óptimo
- o algoritmo exacto (e complexidade) depende da função de penalização de buracos

Alinhamentos Igualmente Óptimos

- muitos alinhamentos óptimos podem existir para um par dado de sequências
- podemos usar escolha de preferências sobre caminhos quando voltamos para trás:



- O *caminho alto* e o *caminho baixo* mostram os dois alinhamentos óptimos mais diferentes.

Análise do Algoritmo de Programação Dinâmica

- Caminho alto:

```
x:   A   A   A   C
y:   A   G   –   C
```

- Caminho baixo:

```
x:   A   A   A   C
y:   –   A   G   C
```

Análise do Algoritmo de Programação Dinâmica

- Existem

$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \approx \frac{2^{2n}}{\sqrt{\pi n}}$$

  alinhamentos possíveis para 2 sequências de tamanho n

- ie, duas sequências de tamanho 1000 têm $\approx 10^{600}$ alinhamentos possíveis
- mas o algoritmo DP encontra o alinhamento óptimo eficientemente.

Complexidade Computacional

- inicialização: $O(m), O(n)$
- preenchendo o resto da matriz: $O(mn)$
- voltar para trás: $O(m + n)$

- se as duas sequências tiverem o mesmo tamanho, a complexidade computacional é:

$$O(n^2)$$

Alinhamento Local

- Até agora discutimos *alinhamento globais*, onde estamos procurando o melhor emparelhamento de duas sequências desde um fim ao outro
- mais frequentemente, queremos um *alinhamento local*, o melhor alinhamento entre subsequências de x e y

Motivação

- útil para comparar sequências de proteínas que partilham um *motivo* (padrão conservado) ou *domínio* (unidade independente enrolada) mas que diferem no resto
- útil para comparar sequências de DNA que partilham um *motivo* (padrão conservado) mas que diferem no resto
- útil para comparar sequências de proteínas contra *sequências de DNA do genoma* (longos grupos de sequências não caracterizadas)
- mais preciso para comparar sequências que divergiram muito

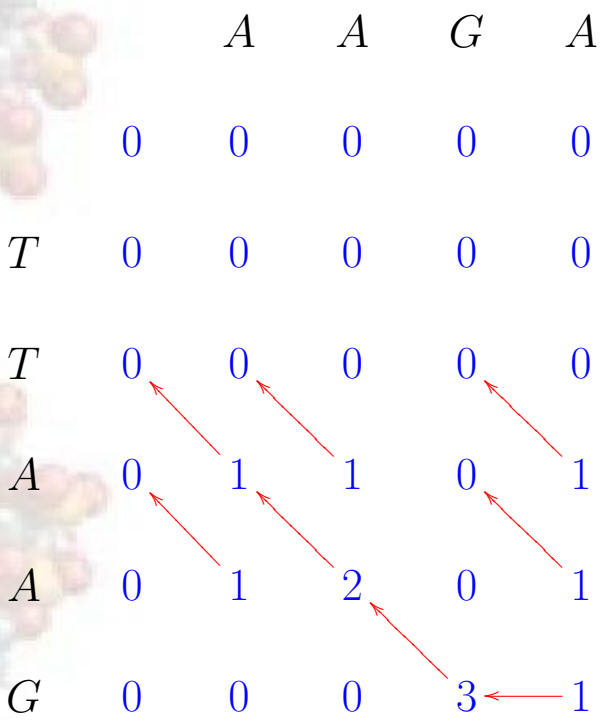
Algoritmo de Alinhamento Local por DP

- formulação original: Smith & Waterman, *Journal of Moelecular Biology*, 1981
- Interpretação das matrizes é um pouco diferente:
  - $F(i, j) =$ pontuação do melhor alinhamento de um sufixo de $x[1 \ldots i]$ e um sufixo de $y[1 \ldots j]$

Algoritmo de Alinhamento Local por DP

- Inicialização: primeira linha e coluna inicializada com 0s
- Retorno:
  - ⋆ encontrar valor máximo de $F(i, j)$; pode ser em qualquer posição da matriz
  - ⋆ parar quando encontrar uma célula com o valor 0.

Exemplo de Alinhamento Local



x: A A G

y: A A G