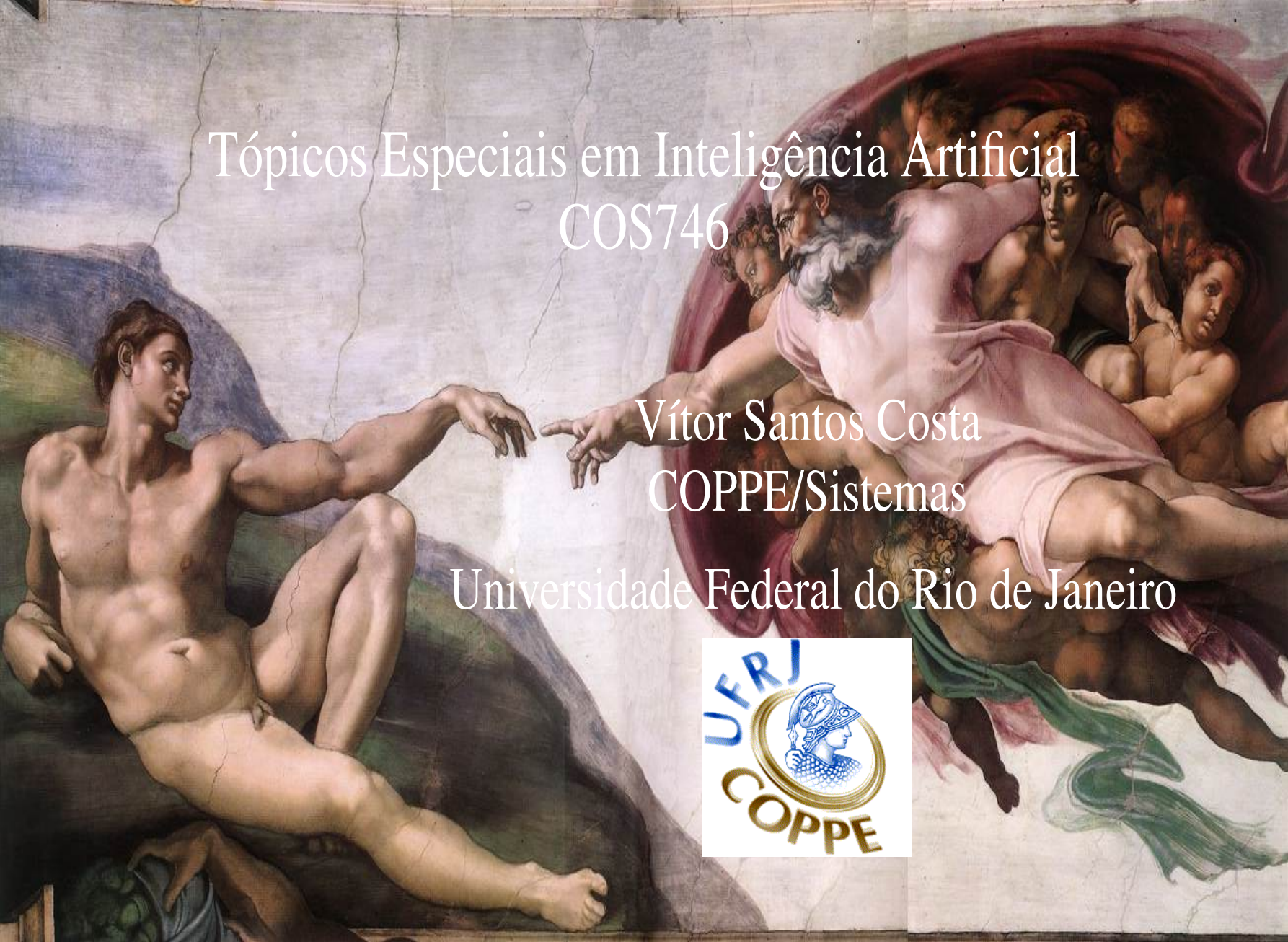


Tópicos Especiais em Inteligência Artificial
COS746

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

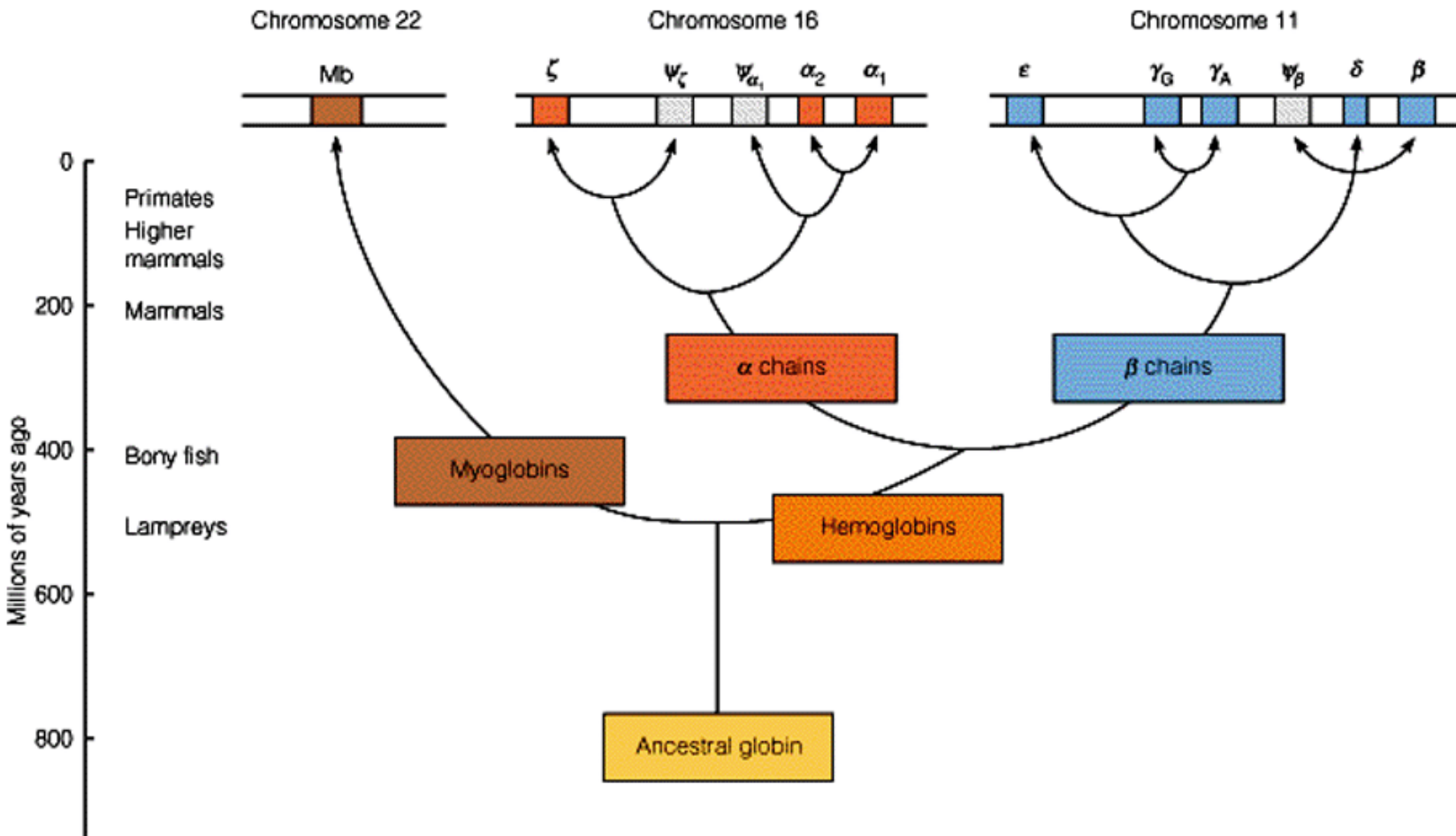
Agradecimento

- Copiado dos slides de Mark Craven/C. David Page para BMI/CS 576, UW-Madison

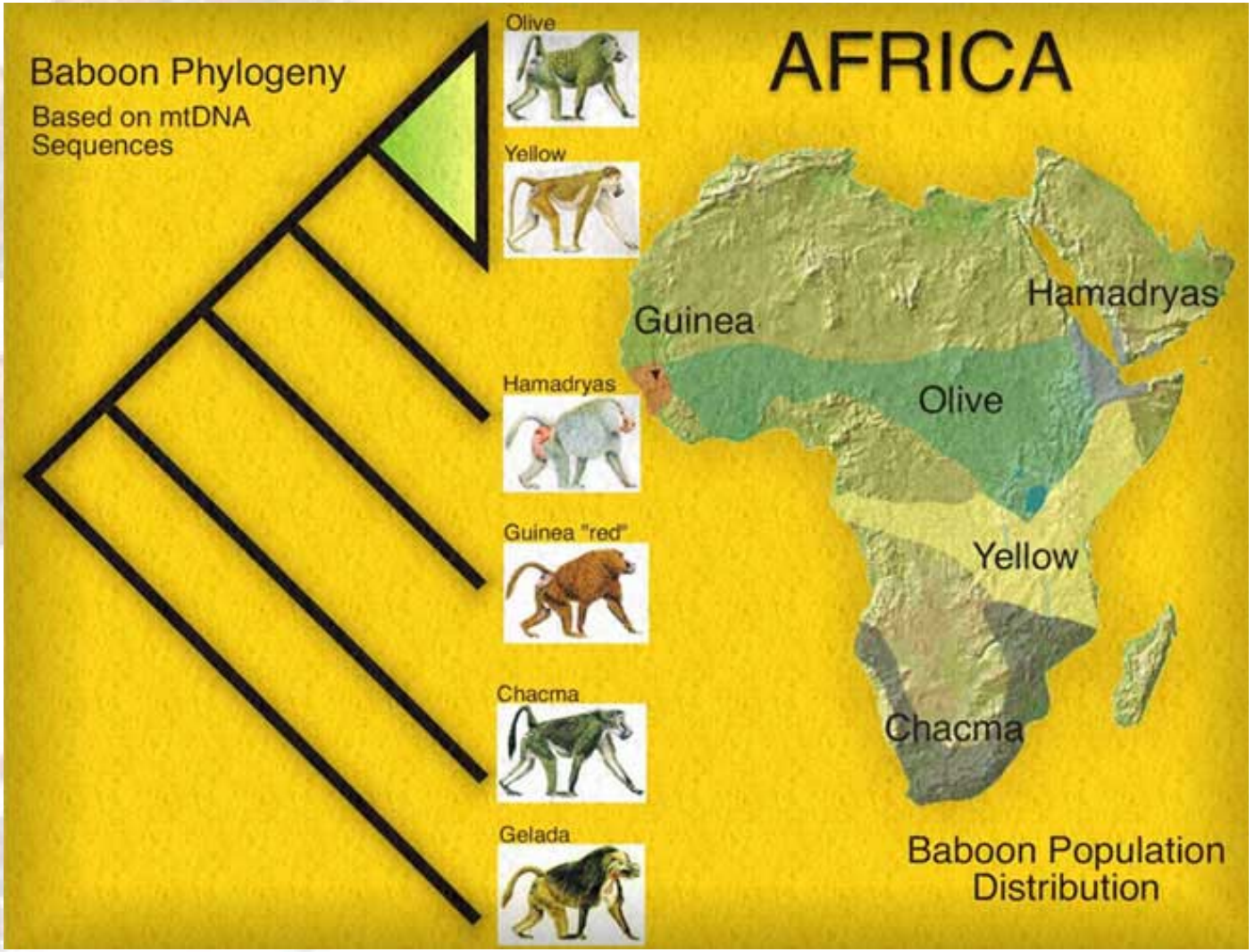
Árvores Filogenéticas

- *Árvore Filogenética*: diagrama mostrando a linha evolucionaria de espécies ou de genes
- Porquê usar árvores:
  - ⋆ para entender a ascendência de várias espécies
  - ⋆ para compreender como várias funções evoluíram
  - ⋆ para informar sobre alinhamentos múltiplos

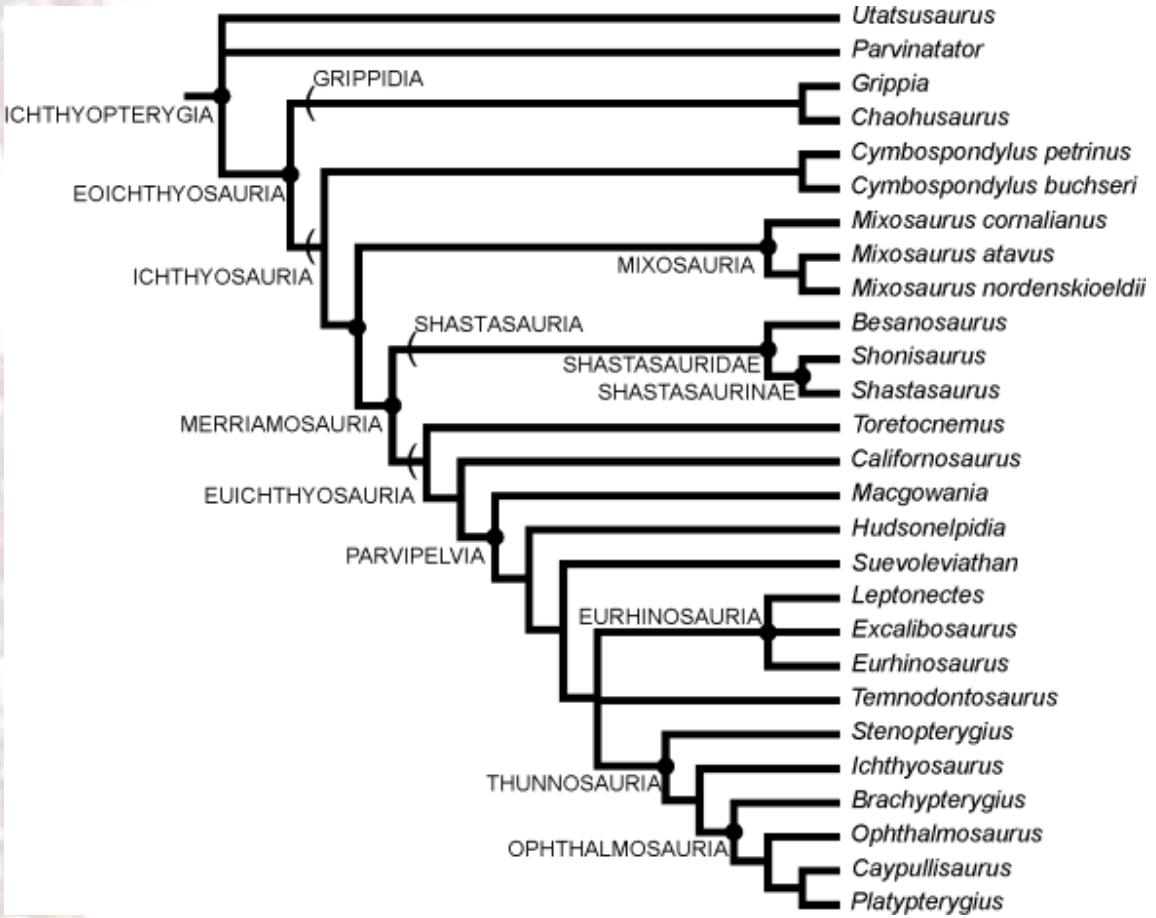
Globin evolution and expression



Exemplo de Filogenia: Babuínos



Exemplo de Filogenia: Ichtiousaurus



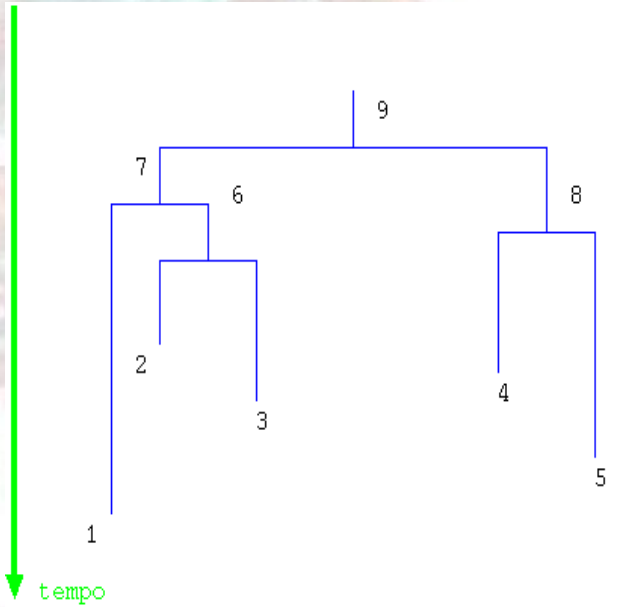
Árvores Filogenéticas: Ideias Básicas

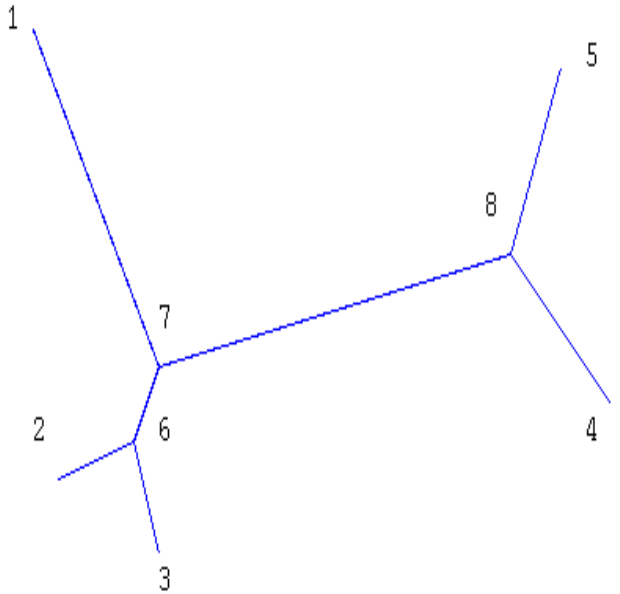
- Folhas representam coisas (genes, indivíduos/famílias, espécies) sendo comparadas
  - ⋆ o termo *taxão* é usado para referir a esses elementos quando representam espécies e classificações mais amplas de organismos
  - ⋆ vamos chamá-las de sequências
- nós internos são hipotéticos antepassados
- numa árvore enraizada, um caminho desde a raíz até a um nó representa um caminho evolucionário
- uma árvore não-enraizada representa relações entre coisas, mas não caminhos evolucionários

Dados para Construir Árvores

- Árvores podem ser construídas de vários tipos de dados:
  - ⋆ *baseados em distâncias*: medidas de distâncias entre espécies/genes
  - ⋆ *baseados em caracteres*: traços morfológicos (eg, pernas), sequências de DNA/proteínas
  - ⋆ *ordem de genes*: ordem linear de genes ortológicos encontrados em genomas dados

Árvores Enraizadas e Não-Enraizadas





Número de Árvores Possíveis

- dadas n sequências, existem $\prod_{i=3}^{n}(2i-5)$ árvores não-enraizadas possíveis
- e $(2n-3)\prod_{i=3}^{n}(2i-5)$ árvores enraizadas

Número de Árvores Possíveis

# sequências (n)	# árvores não-enraizadas	# árvores enraizadas
4	3	15
5	15	105
6	105	945
8	10,395	135,135
10	2,027,025	34,459,425

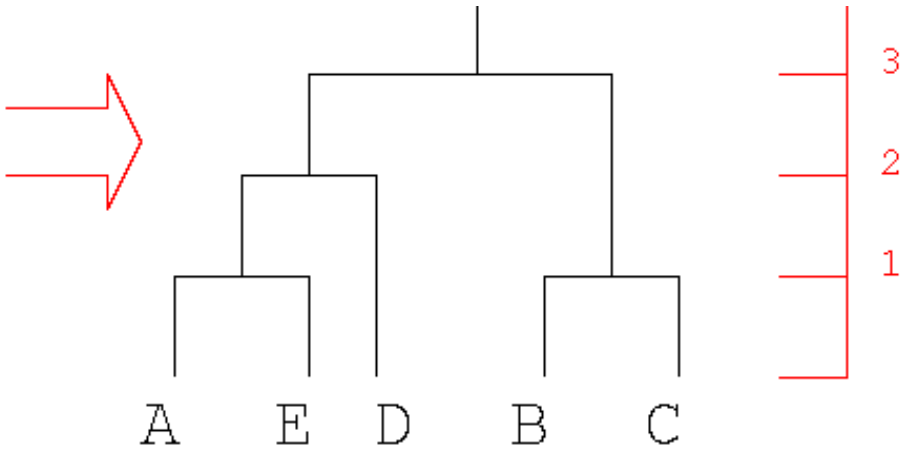
Construção de Árvores Filogenéticas

- Três tipos de métodos gerais:
  - ⋆ *distância*: encontrar uma árvore que explique as distâncias evolucionárias estimadas
  - ⋆ *parcimónia*: encontrar a árvore que requer o número mínimo de alterações para explicar os dados
  - ⋆ *máxima verosimilhança*: encontrar uma árvore que maximize a verosimilhança dos dados

Métodos Baseados em Distância

- **Dados:** uma matriz $n \times n$ M onde M_{ij} é a distância entre os objectos i e j
- **faça:** construa uma árvore pesada nas arestas tal que a distância entre as folhas i e j corresponda a M_ij

	A	B	C	D	E
A	0	8	8	5	3
B		0	3	8	8
C			0	8	8
D				0	5
E					0



O Método UPGMA

- Unweighted Pair Group Method using Arithmetic Averages
- Ideia básica:
  - ⋆ Iterativamente tirar duas sequências/clusters e agregá-los
  - ⋆ criar novo nó na árvore para o cluster agregado
- a distância d_{ij} entre os clusters C_i e C_j de sequências é definida como:

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$$

ou distância média entre pares de sequências de cada cluster

Algoritmo UPGA

- Dar a cada sequência o seu próprio cluster
- definir uma folha para cada sequência e colocar na altura 0
- enquanto há mais de 2 clusters:
  - ⋆ determinar dois clusters i, j com o menor d_{ij}
  - ⋆ defina um novo cluster $C_k = C_i \cup C_j$
  - ⋆ defina um nó k com filhos i e j, coloque-o na altura $d_{ij}/2$
  - ⋆ substitua os clusters i e j com k
- junte os últimos dois clusters, i e j, pela raíz na altura $d_{ij}/2$

UPGMA

- dado um novo cluster C_k formado pela agregação de C_i e de C_j
- podemos calcular a distância entre C_k e qualquer outro cluster C_i como segue:

$$d_{kl} = \frac{d_{il}|C_i| + d_{jl}|C_j|}{|C_i| + |C_j|}$$

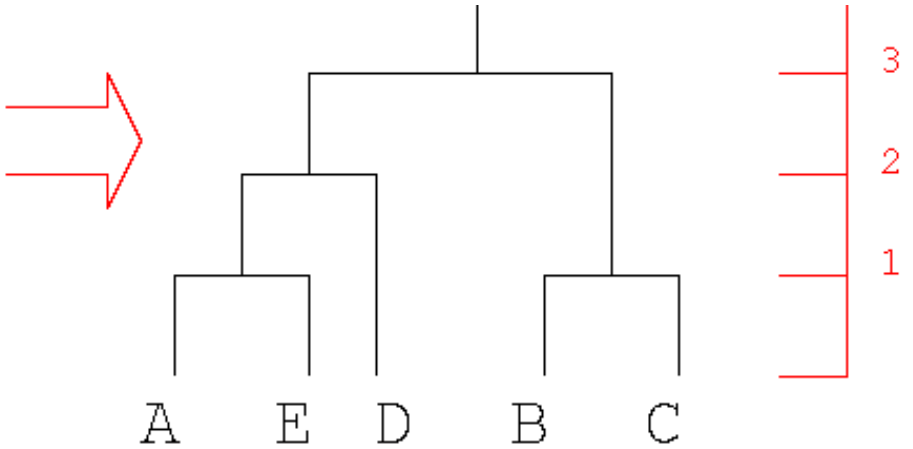
A Premissa do Relógio Molecular e Dados Ultramétricos

- A *premissa do relógio molecular*: divergência das sequências é assumida ocorrer à mesma velocidade em todos os pontos da árvore
- esta premissa não é verdade em geral: pressões evolucionárias variam de acordo com o tempo, organismos, genes num organismo e regiões num gene
- se podemos assumir esta premissa, os dados são chamados de *ultramétricos*

Dados Ultramétricos: Condição Necessária e Suficiente

- Dados Ultramétricos: para qualquer tripla de sequências i, j, k as distâncias ou são todas iguais, ou duas são iguais e a restante é menor.

	A	B	C	D	E
A	0	8	8	5	3
B		0	3	8	8
C			0	8	8
D				0	5
E					0



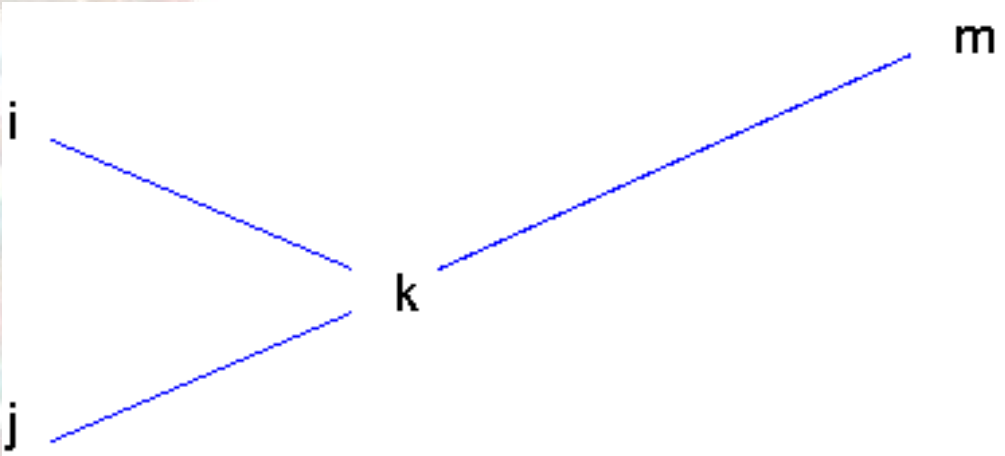
Junção de Vizinhos

- com em UPGMA, construímos uma árvore juntando iterativamente sub-árvores
- diferente de UPGMA:
  - ⋆ não assumimos o relógio molecular
  - ⋆ produz árvore não enraizada
- assuma *aditividade*: a distância entre dois pares de folhas é a soma dos comprimentos dos vértices que fazem a ligação.

Distâncias em Junção de Vizinhos

- dado um novo nó interno k, a distância para outro nó m é dada por:

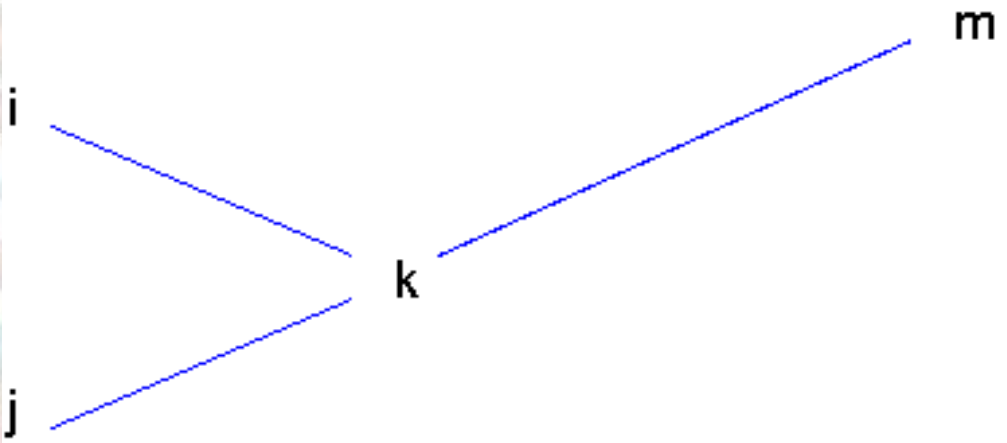
$$d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$$



Distâncias em Junção de Vizinhos

- Podemos calcular a distância de uma folha para o nó pai na seguinte forma:

$$d_{ik} = \frac{1}{2}(d_{ij} + d_{im} - d_{jm})$$



$$d_{jk} = d_{ij} - d_{ik}$$

Distâncias em Junção de Vizinhos

- Podemos generalizar esta regra de forma a tomar em conta a distância para todas as outras folhas:

$$d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$$

onde

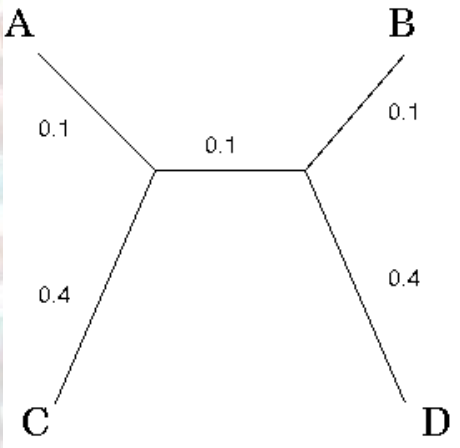
$$r_i = \frac{1}{|L| - 2} \sum_{m \in L} d_{im}$$

e L é o conjunto das folhas

- isto é mais robusto se os dados não forem estritamente aditivos

Juntar que Nós?

- Em cada passo escolhemos um par de nós para juntar. Devemos escolher os nós com o menor d_{ij}?
- Suponhamos que a árvore verdadeira parece como isto e que estamos a escolher os primeiros nós para juntar:



$$d_{AB} = 0.3$$
$$d_{AC} = 0.5$$

- Decisão errada em juntar A e B: precisamos de considerar distância do par até outras folhas.

Juntar que Nós?

- Para evitar o problema escolha o par de nós baseado nas distâncias baseado em D_{ij}:

$$D_{ij} = d_{ij} - (r_i + r_j)$$

$$r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}$$

Algoritmo de Junção de Vizinhos

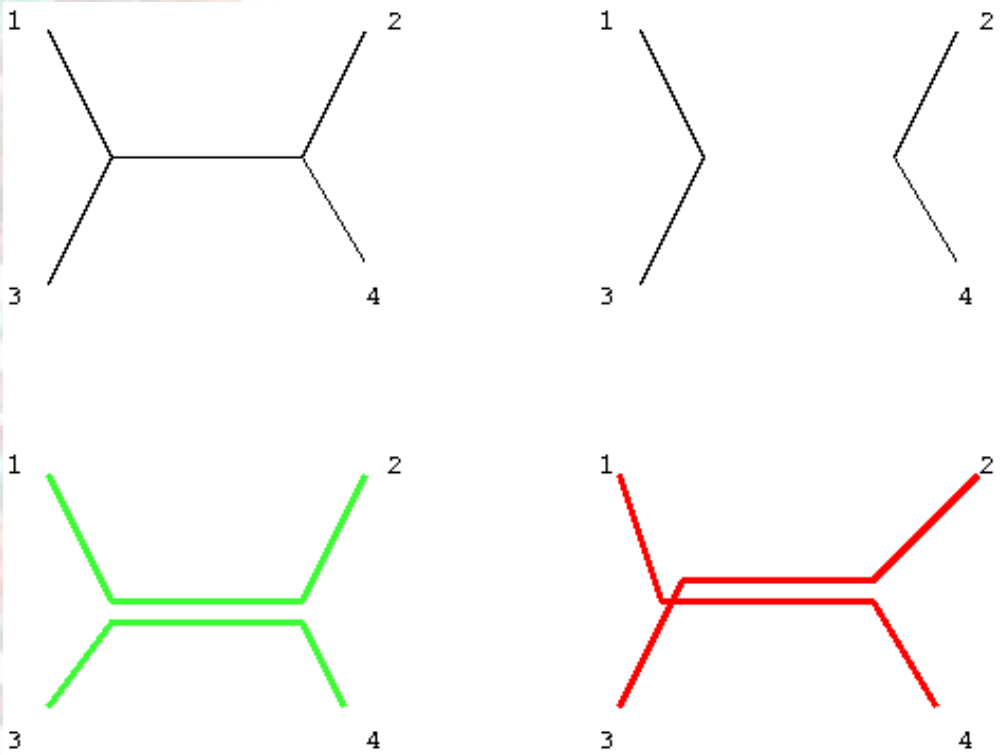
- defina a árvore T como o conjunto de nós folhas
- $L = T$
- enquanto há mais que duas sub-árvores em T:
  - ⋆ escolha o par i, j em L com D_{ij} mínimo
  - ⋆ adicione a T um novo nó agregando i e j
  - ⋆ determine novas distâncias:

$$d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$$
$$d_{jk} = d_{ij} - dik$$
$$d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij}) \text{para todos os outros} m \in L$$

  - ⋆ remova i e j de L e insira k (processe-o como se uma folha)
- junte as duas árvores restantes, i e j com um vértice de comprimento d_{ij}

Testando Aditividade

- Para qualquer conjunto de qualquer folhas i, j, k, l duas das distâncias $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$ e $d_{il} + d_{jk}$ devem ser iguais e maiores que a terceira distância



Escolhendo Raízes

- Escolher uma raíz para árvores não-enraizadas é muitas vezes feita usando um "outgroup"
- *Outgroup* é uma espécie que se sabe ser mais diferentes das outras espécies do que elas são entre elas.
- o ponto onde o outgroup se junta ao resto da árvore é o melhor candidato para a raíz.

Comentários Sobre Métodos Baseados em Distância

- Se os dados de distância são ultramétricos (e as distâncias são distâncias genuínas), então UPGMA encontra a árvore certa
- Se os dados são aditivos (e as distâncias são distâncias genuínas), então junção de vizinhos identifica a árvore correcta
- senão, os métodos podem não recuperar a árvore correcta, mas são boas heurísticas

Construção de Árvores Filogenéticas

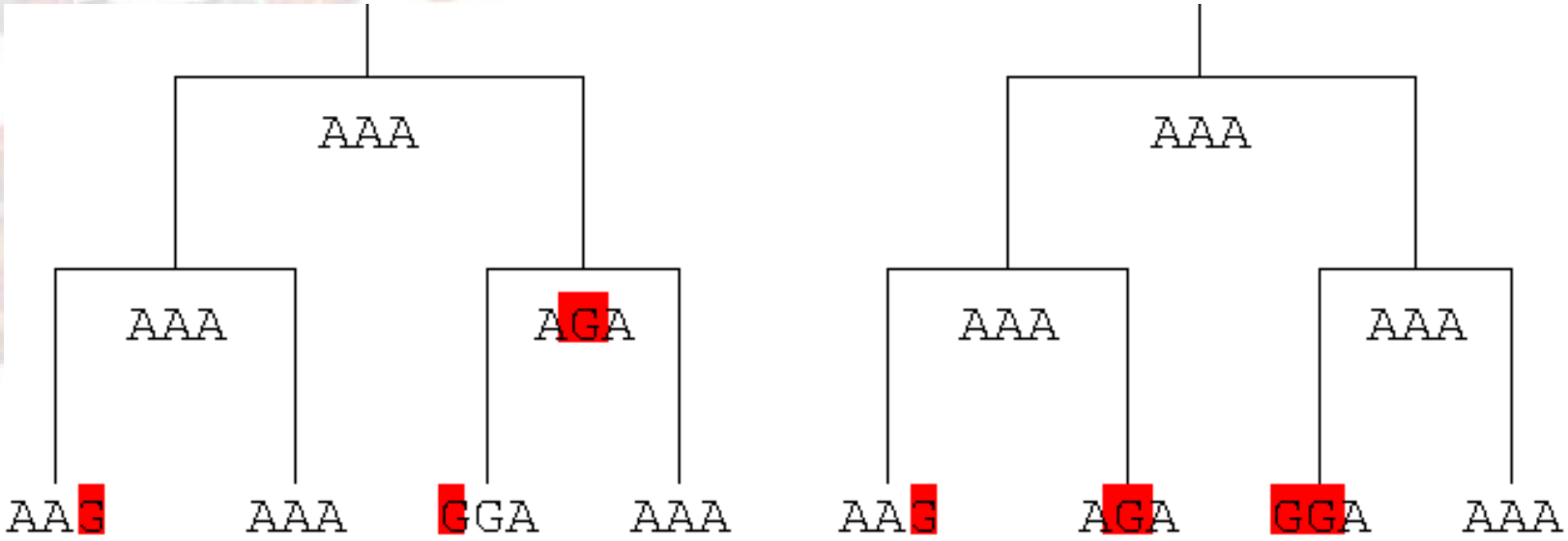
- Três tipos de métodos gerais:
  - ⋆ *distância*: encontrar uma árvore que explique as distâncias evolucionárias estimadas
  - ⋆ *parcimónia*: encontrar a árvore que requer o número mínimo de alterações para explicar os dados
  - ⋆ *maximum likelihood*: encontrar uma árvore que maximize a verosimilhança dos dados

Métodos Baseados em Parcinómia

- *dados*: dados baseados em caracteres
- *faça*: encontrar árvore que explique os dados com o número mínimo de alterações.

Exemplo de Parcinómia

- existem muitas árvores que podem explicar a filogenia das sequências seguintes: AAG, AAA GGA, AGA.



- parcimónia prefere a primeira árvore porque requer menor número de substituições

Métodos Baseados em Parcimónia

- habitualmente estes métodos envolvem dois componentes:
  - ⋆ uma procura pelo espaço das árvores
  - ⋆ um processamento para explicar o menor número de mudanças necessárias para explicar os dados (para uma dada topologia).

Encontrar Menor Número de Mudanças Numa Árvore

- Algoritmo de Fitch [1971]:
  - ⋆ assume qualquer estado (nucleotído, amino-ácido) e pode converter para qualquer outro estado
  - ⋆ assume que as posições são independentes

Algoritmo de Fitch

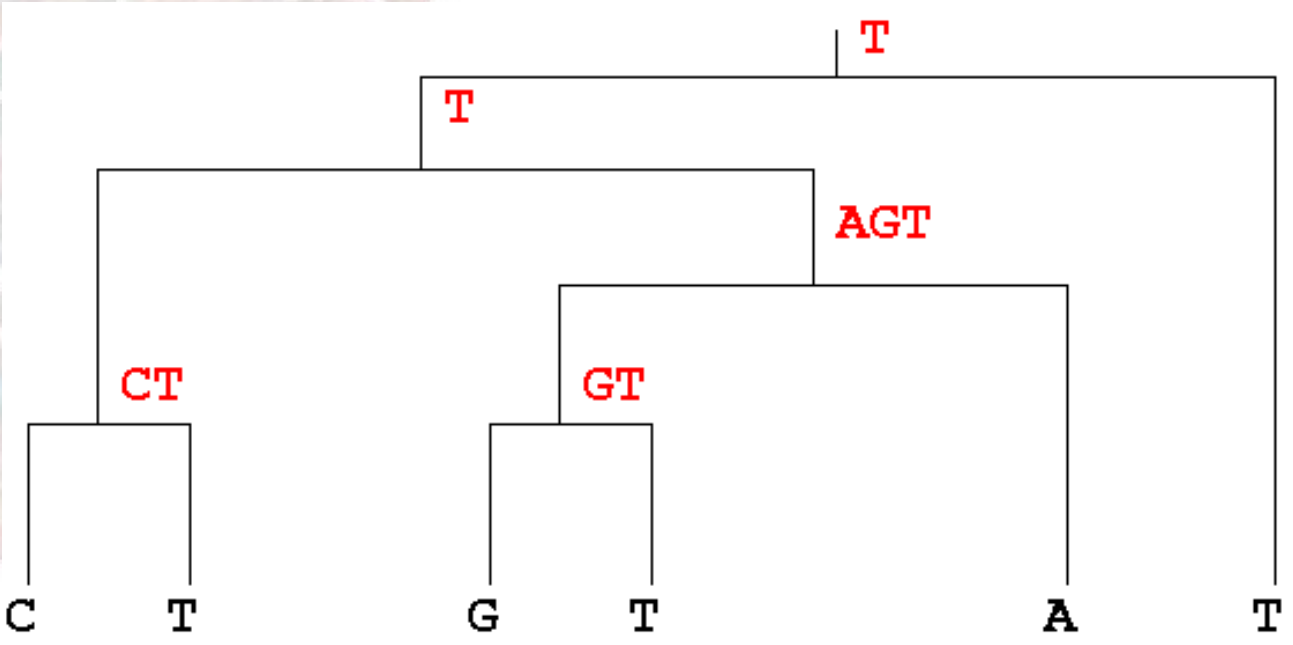
- atravessa a árvore desde as folhas até à raíz determinando o número possível de *estados* (eg, nucleotídos) que podem ser tomados por cada nó interno.
- atravessa a árvore desde a raíz até às folhas estabelecendo os estados para os nós internos.

Passo 1: Estado Possível para os Nós Internos

- atravesse a árvore em pós-ordem (desde as folhas até à raíz)
- determinar os estados possíveis R_i do nó interno i com filhos j e k:

$$R_i = \begin{cases} R_j \cup R_k, & \text{se} R_j \cap R_k = \emptyset \\ R_j \cap R_k, & \text{senão} \end{cases}$$

O Algoritmo de Fitch: Passo 1



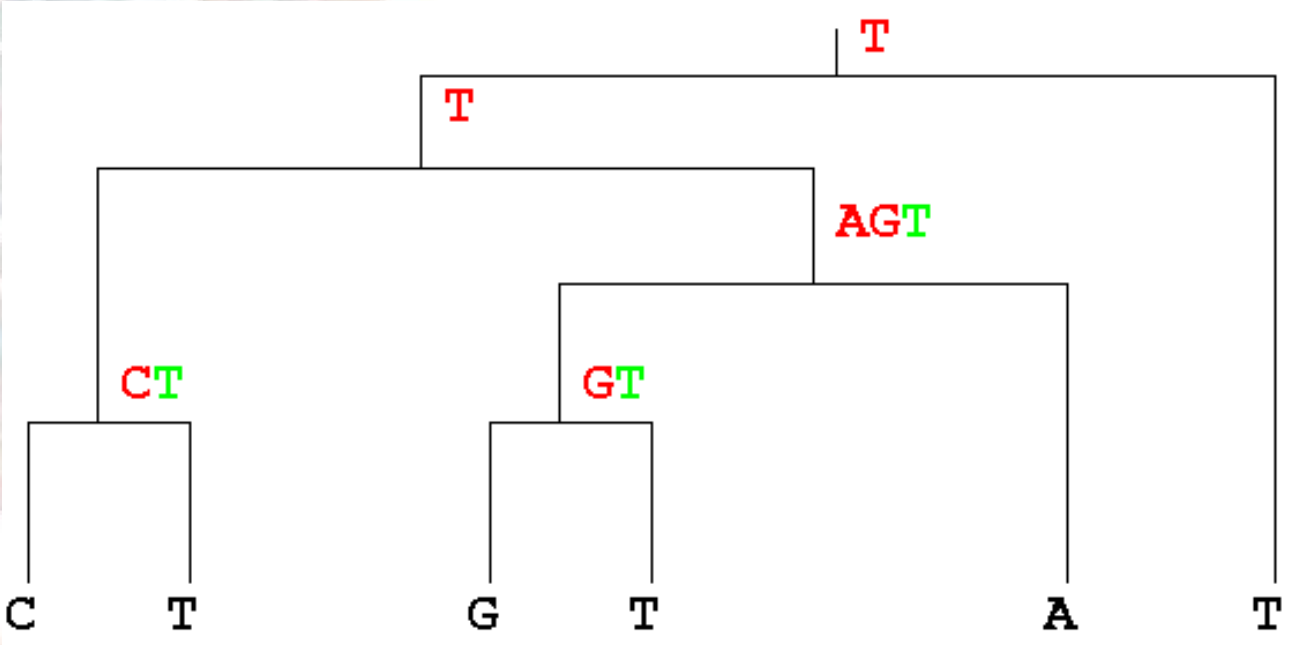
- # de mudanças = # de uniões

O Algoritmo de Fitch: Passo 2

- atravesse a árvore em pré-ordem (desde a raíz até às folhas)
- seleccionar um estado r_j do nó interno j com pai i:

$$r_j = \begin{cases} r_i, & \text{se} r_i \in R_j \\ \text{estado arbitrário} \in R_j, & \text{senão} \end{cases}$$

O Algoritmo de Fitch: Passo 2



O Algoritmo de Fitch com Pesos

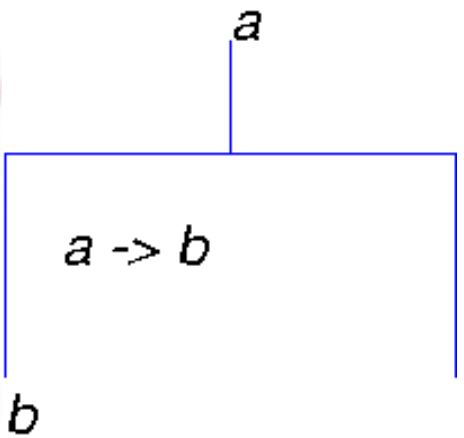
- Sankoff & Cedergren [1983]
- Em vez de assumir que todos as mudanças de estado são igualmente prováveis, use custos diferentes $S(a, b)$ para mudanças diferentes
- primeiro passo do algoritmo é propagar custos subindo na árvore:

$$a \longrightarrow b$$

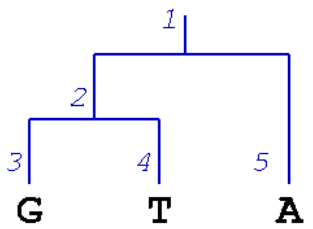
O Algoritmo de Fitch com Pesos

- para um nó interno i com filhos j e k

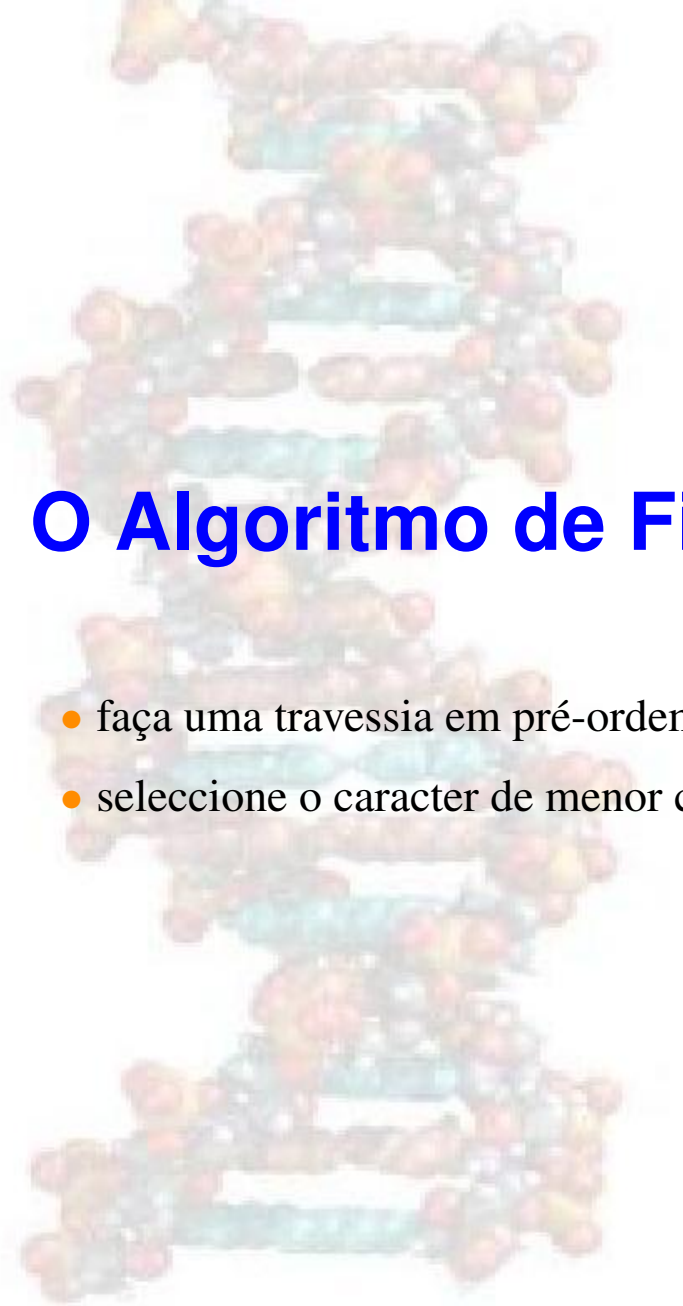
$$R_i(a) = min_b(R_j(b) + S(a,b)) + min_b(R_k(b) + S(a,b))$$



O Algoritmo de Fitch com Pesos

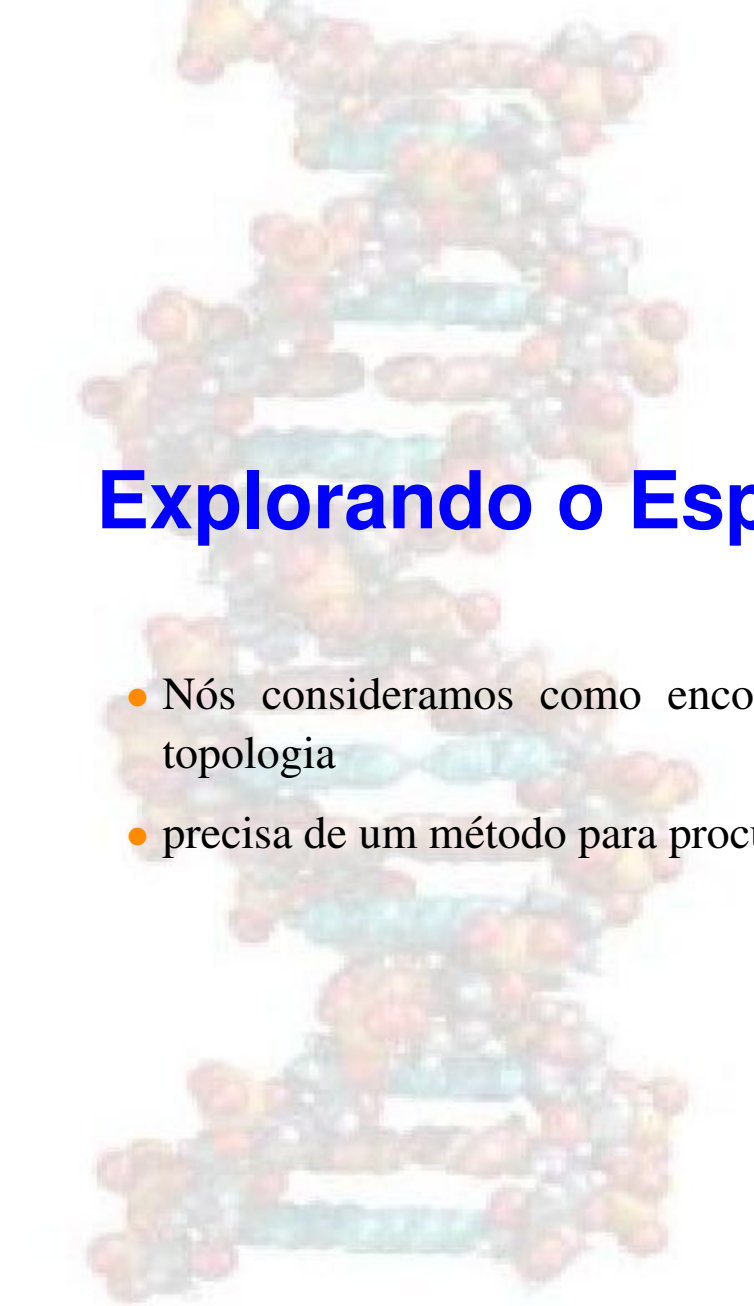


- $R_3[\mathtt{A}] = \infty, R_3[\mathtt{C}] = \infty, R_3[\mathtt{G}] = 0, R_3[\mathtt{T}] = \infty$
- $R_4[\mathtt{A}] = \infty, R_4[\mathtt{C}] = \infty, R_4[\mathtt{G}] = \infty, R_4[\mathtt{T}] = 0$
- $\begin{cases} R_2[\mathtt{A}] = R_3[\mathtt{G}] + S(\mathtt{A},\mathtt{G}) + R_4[\mathtt{T}] + S(\mathtt{A},\mathtt{T}) \\ \ldots \\ R_2[\mathtt{T}] = R_3[\mathtt{G}] + S(\mathtt{A},\mathtt{T}) + R_4[\mathtt{T}] + S(\mathtt{T},\mathtt{T}) \end{cases}$
- $R_5[\mathtt{A}] = 0, R_5[\mathtt{C}] = \infty, R_5[\mathtt{G}] = \infty, R_5[\mathtt{T}] = \infty$
- $\begin{cases} R_1[\mathtt{A}] = min(R_2[\mathtt{A}] + S(\mathtt{A},\mathtt{A}), \ldots, R_2[\mathtt{T}] + S(\mathtt{A},\mathtt{T})) + R_5[\mathtt{A}] + S(\mathtt{A},\mathtt{A}) \\ \ldots \\ R_1[\mathtt{T}] = min(R_2[\mathtt{A}] + S(\mathtt{T},\mathtt{A}), \ldots, R_2[\mathtt{T}] + S(\mathtt{T},\mathtt{T})) + R_5[\mathtt{A}] + S(\mathtt{A},\mathtt{T}) \end{cases}$



O Algoritmo de Fitch com Pesos: Passo 2

- faça uma travessia em pré-ordem da árvore (desde a raíz para as folhas)
- seleccione o caracter de menor custo para cada nó



Explorando o Espaço das Árvores

- Nós consideramos como encontrar o menor número de mudanças para cada topologia
- precisa de um método para procurar no espaço das árvores

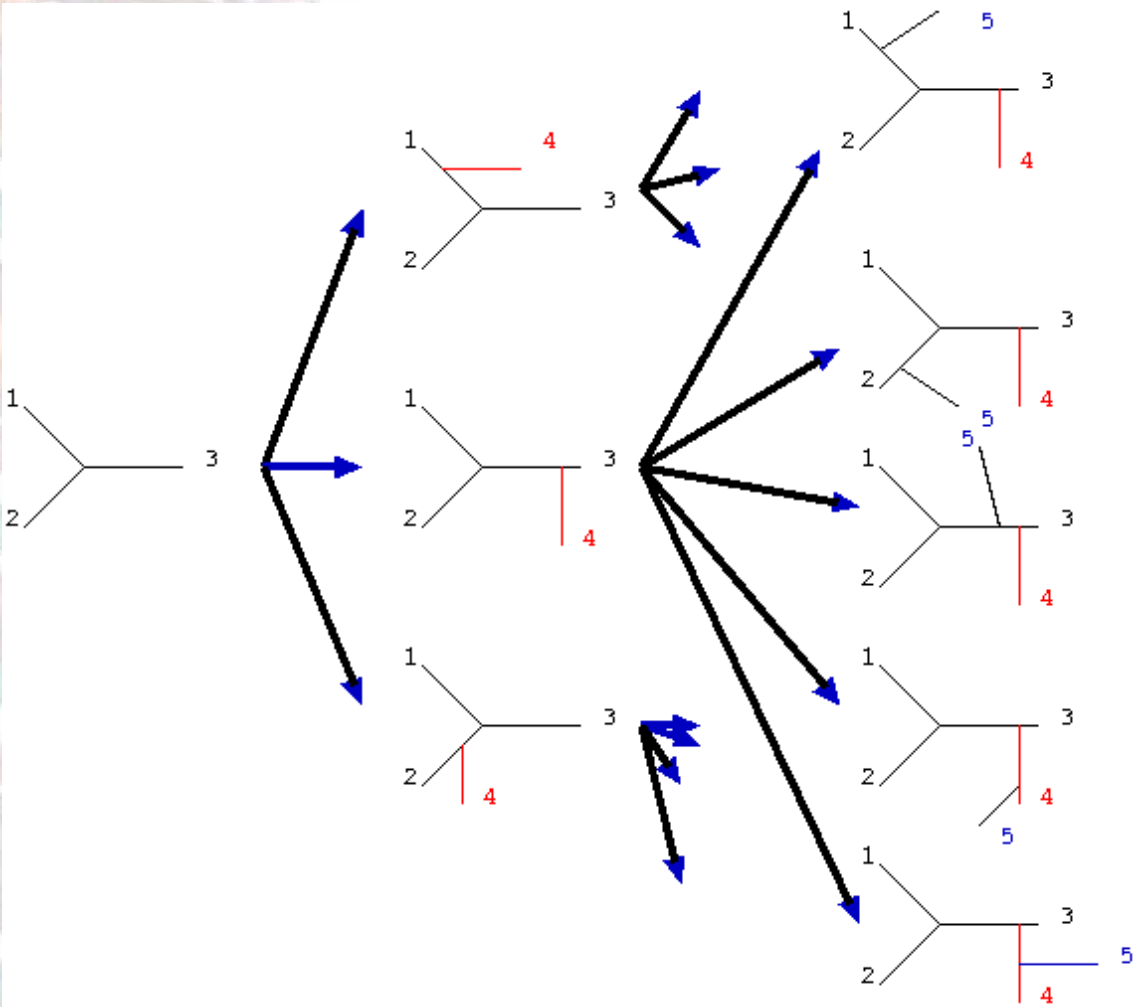
Métodos de Procura

- exaustiva
- branch & bound:
  - ⋆ encontre um árvore inicial (eg, por UPGMA ou por junção de vizinhos) e determine o custo
  - ⋆ use procura para encontrar outras árvores:
    - ∗ abandone árvores parciais cujo custo excede a árvore de menor custo até agora
- métodos gulosos: eg, troca de ramos

Procura Por Branch & Bound

- procure pelo espaço das árvores sem raíz:
  - ⋆ adicione folhas à árvore incrementalmente
  - ⋆ mantenha a árvore de custo menor completa até agora T'
  - ⋆ corte uma árvore T e os seus descendentes se $custo(T) > custo(T')$
- Propriedade Chave: adicionar folhas só pode aumentar o custo da árvore

Procura Por Branch & Bound



Algoritmo

- Para n sequências mantenha um vector de contadores:

$$[i_3][i_5][i_7]\ldots[i_{2n-5}]$$

  onde i_k toma os valores $0 \ldots k$
- uma árvore completa é representada por uma atribuição de todos os i_k a valores não-zero.
- i_k indica, com uma árvore parcial com k vértices, onde adicionar um ramo para a sequência seguinte
- $i_k = 0$ indica uma árvore parcial

Algoritmo

- Para procurar o espaço, rode contadores através dos seus valores possíveis (como se fossem odómetros):
  - ⋆ contadores mais à direita mudam mais depressa
  - ⋆ quando um contador é zero, os contadores à direita devem ser 0 também
  - ⋆ teste o custo (parcial) da árvore em cada tick
  - ⋆ faça com que o odómetro salte quando há um corte

Algoritmo

- É um método completo
  - ⋆ garantido encontrar solução óptima
- frequentemente muito mais eficiente que procura exaustiva
- no pior caso, não é melhor
- a eficiência depende da qualidade da árvore inicial

Comentários sobre Inferência de Árvores

- o espaço de procura pode ser grande, mas pode encontrar a árvore óptima eficientemente em alguns casos
- em alguns casos métodos heurísticos podem ser aplicados
- difícil avaliar filogenias inferidas: a verdade-alvo não é habitualmente sabida:
  - ⋆ podemos olhar para a concordância entre diferentes fontes de evidência
  - ⋆ quando a procura não é completa, podemos procurar repetibilidade em sub-amostras dos dados
- alguns métodos novos usam dados baseados na ordem linear dos genes ortológicos no cromossoma
- filogenia de bactérias e vírus não é trivial devido a transferências laterais de material genético: *filogenias locais* podem ser mais apropriadas

Comentários sobre Inferência de Árvores

Um visão diferente:

- http://evolution.genetics.washington.edu/genet541/2002/lecture1.pdf
- Phylip: http://evolution.genetics.washington.edu/phylip.html
- MrBayes: http://mrbayes.csit.fsu.edu/