Bioinformatics

Lecture 4

Weidong Tian, School of Life Sciences, Fudan University

Sept 26, 2024

Outline

- Review of last lecture
- Phylogenetic tree construction
- Hidden Markov Model

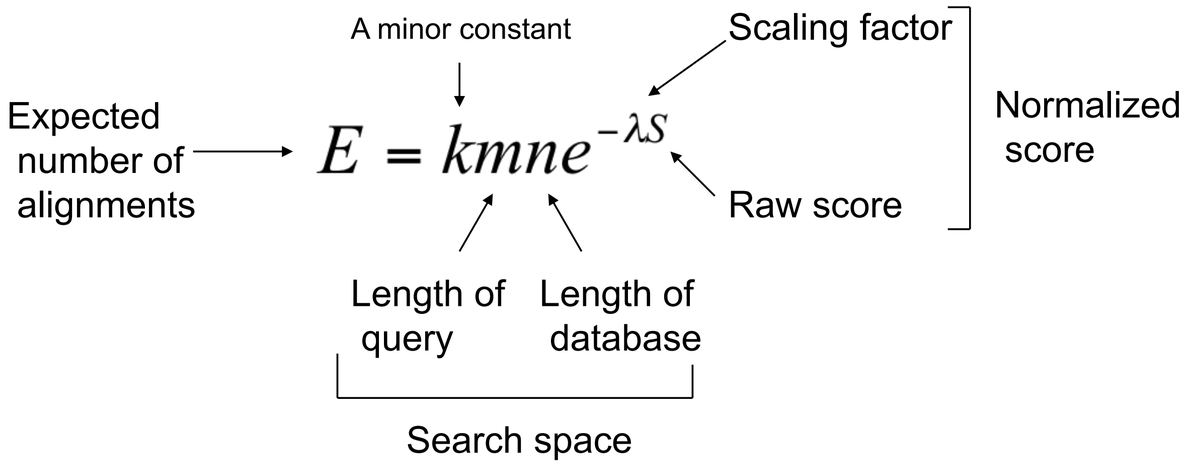
BLAST

- BLAST, the Basic Local Alignment Search Tool (Altschul et al., 1990), is perhaps the most widely used bioinformatics tool ever written. It is an alignment heuristic that determines “local alignments” between a *query* and a *database*. It uses an approximation of the Smith-Waterman algorithm.

- BLAST consists of two components: a search algorithm and computation of the statistical significance of the alignments.

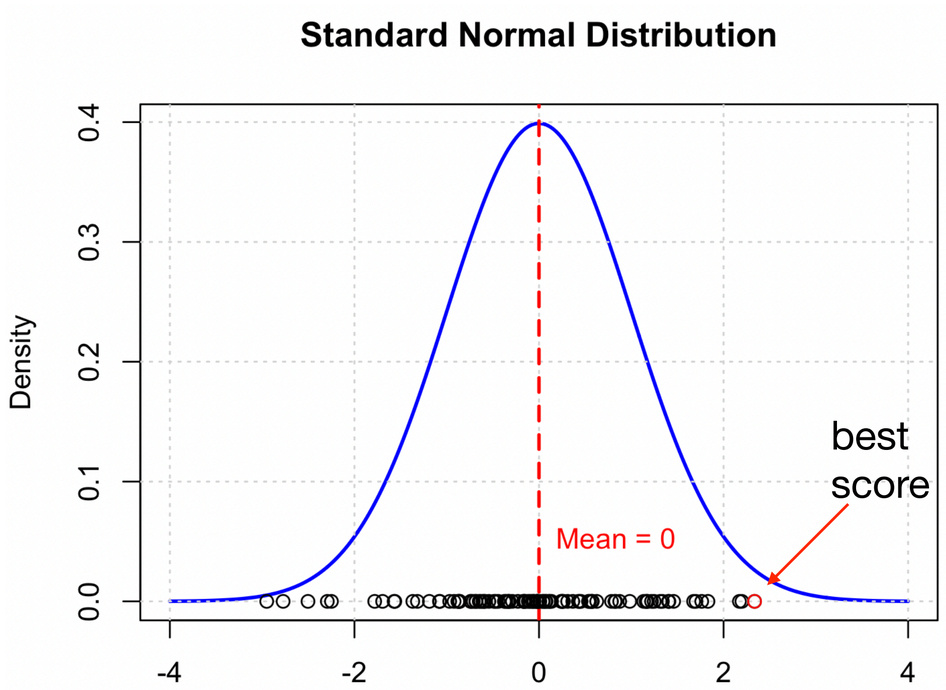
The BLAST E-value

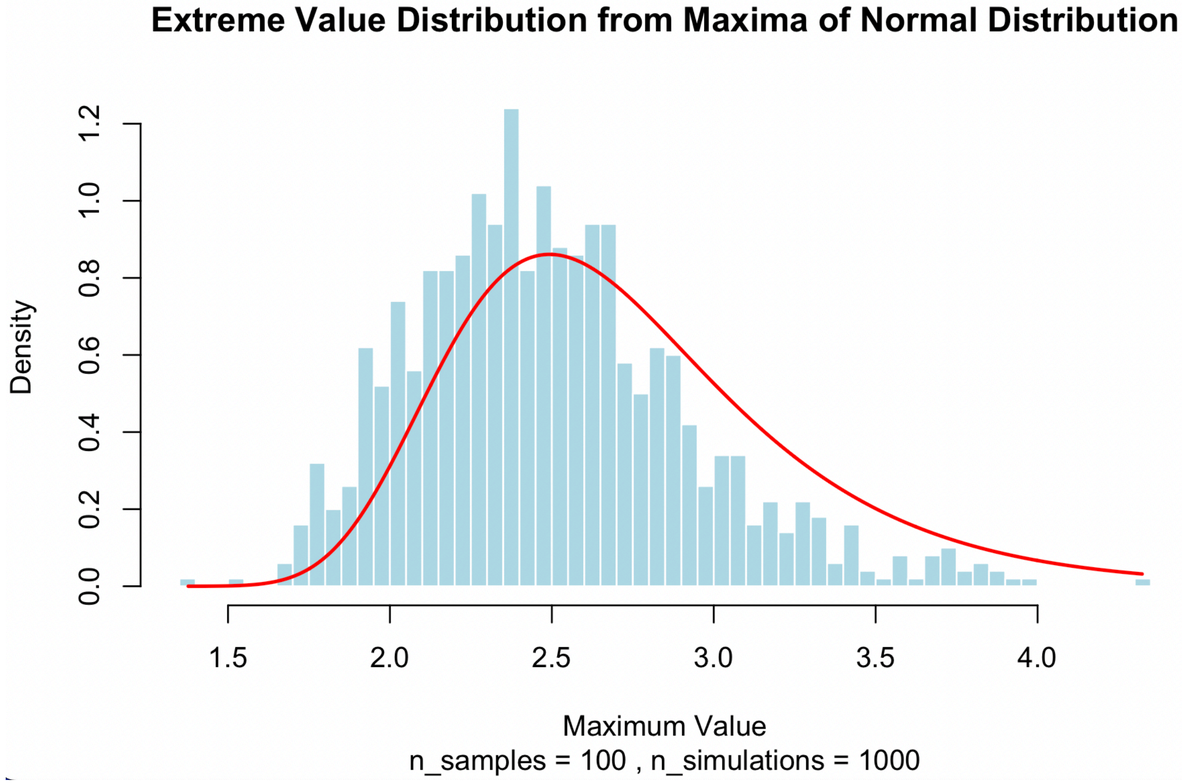
The Karlin-Altschul Equation



- λ is the scaling factor that determines the steepness of the score distribution. It directly affects how quickly the probability of achieving high alignment scores decreases. It is derived from the scoring matrix (e.g., BLOSUM, PAM) and gap penalties used in the alignment process.
- K is related to the search space size and corresponds to the location parameter in EVD. As K increases in BLAST (due to a larger database or longer query sequence), the distribution of alignment scores is shifted to account for the increased probability of finding high-scoring alignments by chance.
- Large e-value (5 or 10) indicates that the alignment is probably by chance.
- E-values of 0.1 or 0.05 are typical cutoff values for database search.

Extreme Value Distribution





- "Randomly sample 100 values from a standard normal distribution (mean = 0, sd = 1) and identify the best score.

- Repeat this process 1000 times. What is the resulting distribution of these best scores?"

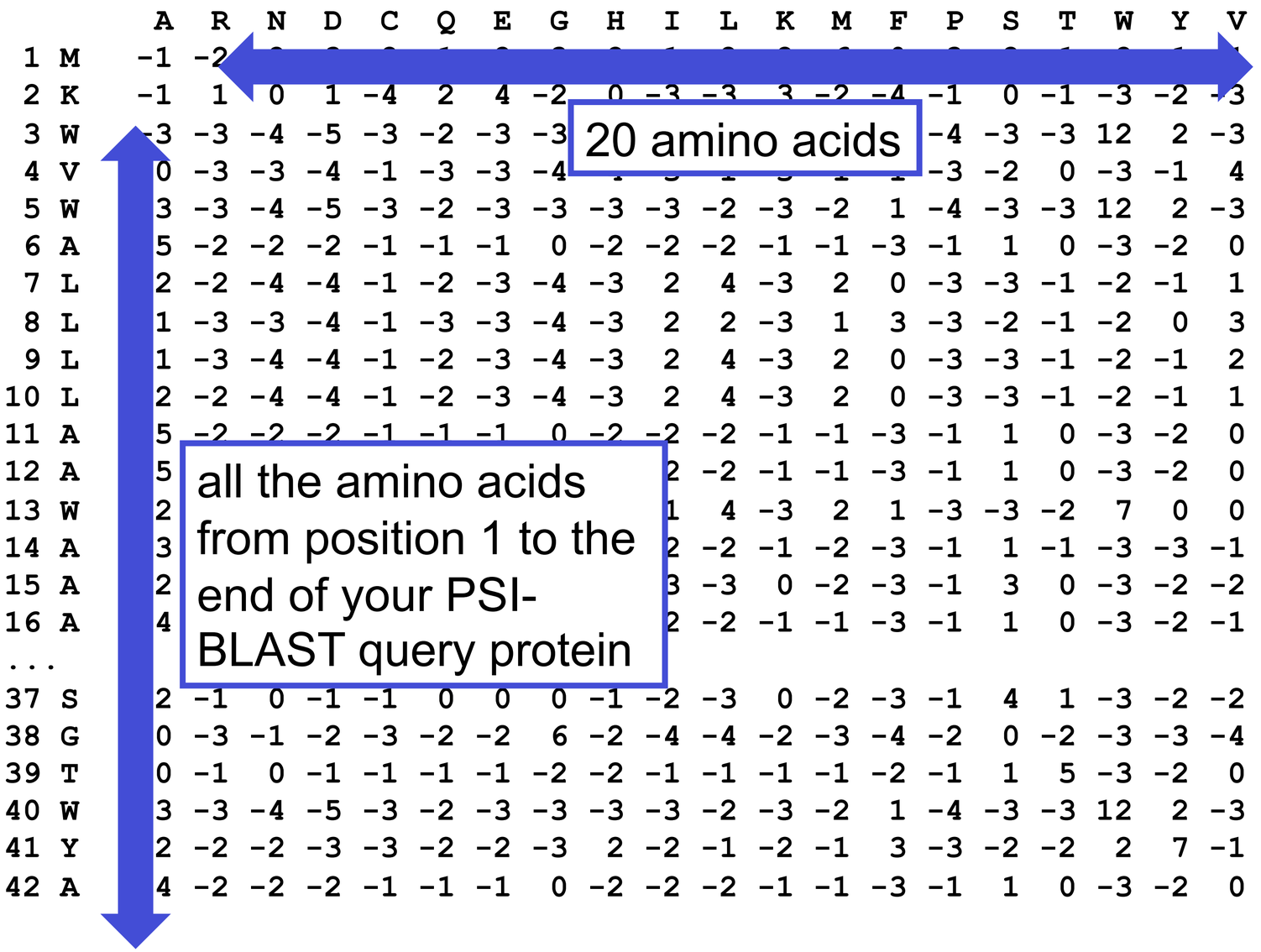
Raw score vs. bit score

$$S' = \frac{\lambda S - \ln K}{\ln 2}$$

$$E = mn\,2^{-S'}$$

- Raw scores: The raw score is calculated directly from the alignment of two sequences using the substitution matrix (such as BLOSUM62 or PAM) and any penalties for gaps (insertions or deletions).
  - The raw score is dependent on the scoring matrix and gap penalties used.
  - It does not adjust for the size of the database or query sequence.
  - A higher raw score indicates a better alignment.
- Bit scores: The bit score is a normalized version of the raw score that accounts for the statistical properties of the scoring system, including the size of the search space. This makes bit scores comparable across different searches, even if they use different databases, scoring matrices, or gap penalties.

Example of a PSSM



		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
1	M	-1	-2																		
2	K	-1	1	0	1	-4	2	4	-2	0	-3	-3	3	-2	-4	-1	0	-1	-3	-2	-3
3	W	-3	-3	-4	-5	-3	-2	-3	-3							-4	-3	-3	12	2	-3
4	V	0	-3	-3	-4	-1	-3	-3	-4							-3	-2	0	-3	-1	4
5	W	-3	-3	-4	-5	-3	-2	-3	-3	-3	-3	-2	-3	-2	1	-4	-3	-3	12	2	-3
6	A	5	-2	-2	-2	-1	-1	-1	0	-2	-2	-2	-1	-1	-3	-1	1	0	-3	-2	0
7	L	2	-2	-4	-4	-1	-2	-3	-4	-3	2	4	-3	2	0	-3	-3	-1	-2	-1	1
8	L	1	-3	-3	-4	-1	-3	-3	-4	-3	2	2	-3	1	3	-3	-2	-1	-2	0	3
9	L	1	-3	-4	-4	-1	-2	-3	-4	-3	2	4	-3	2	0	-3	-3	-1	-2	-1	2
10	L	2	-2	-4	-4	-1	-2	-3	-4	-3	2	4	-3	2	0	-3	-3	-1	-2	-1	1
11	A	5	-2	-2	-2	-1	-1	-1	0	-2	-2	-2	-1	-1	-3	-1	1	0	-3	-2	0
12	A	5									2	-2	-1	-1	-3	-1	1	0	-3	-2	0
13	W	2									1	4	-3	2	1	-3	-3	-2	7	0	0
14	A	3									2	-2	-1	-2	-3	-1	1	-1	-3	-3	-1
15	A	2									3	-3	0	-2	-3	-1	3	0	-3	-2	-2
16	A	4									2	-2	-1	-1	-3	-1	1	0	-3	-2	-1
...																					
37	S	2	-1	0	-1	-1	0	0	0	-1	-2	-3	0	-2	-3	-1	4	1	-3	-2	-2
38	G	0	-3	-1	-2	-3	-2	-2	6	-2	-4	-4	-2	-3	-4	-2	0	-2	-3	-3	-4
39	T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-3	-2	0
40	W	3	-3	-4	-5	-3	-2	-3	-3	-3	-3	-2	-3	-2	1	-4	-3	-3	12	2	-3
41	Y	2	-2	-2	-3	-3	-2	-2	-3	2	-2	-1	-2	-1	3	-3	-2	-2	2	7	-1
42	A	4	-2	-2	-2	-1	-1	-1	0	-2	-2	-2	-1	-1	-3	-1	1	0	-3	-2	0

Position-Specific Iterated BLAST (PSI-BLAST)

(1) PSI-BLAST takes as an input a single protein sequence and compares it to a protein database, using the gapped BLAST program

(2) The program constructs a multiple alignment, and then a profile, from any significant local alignments found. The original query sequence serves as a template for the multiple alignment and profile, whose lengths are identical to that of the query. Different numbers of sequences can be aligned in different template positions.

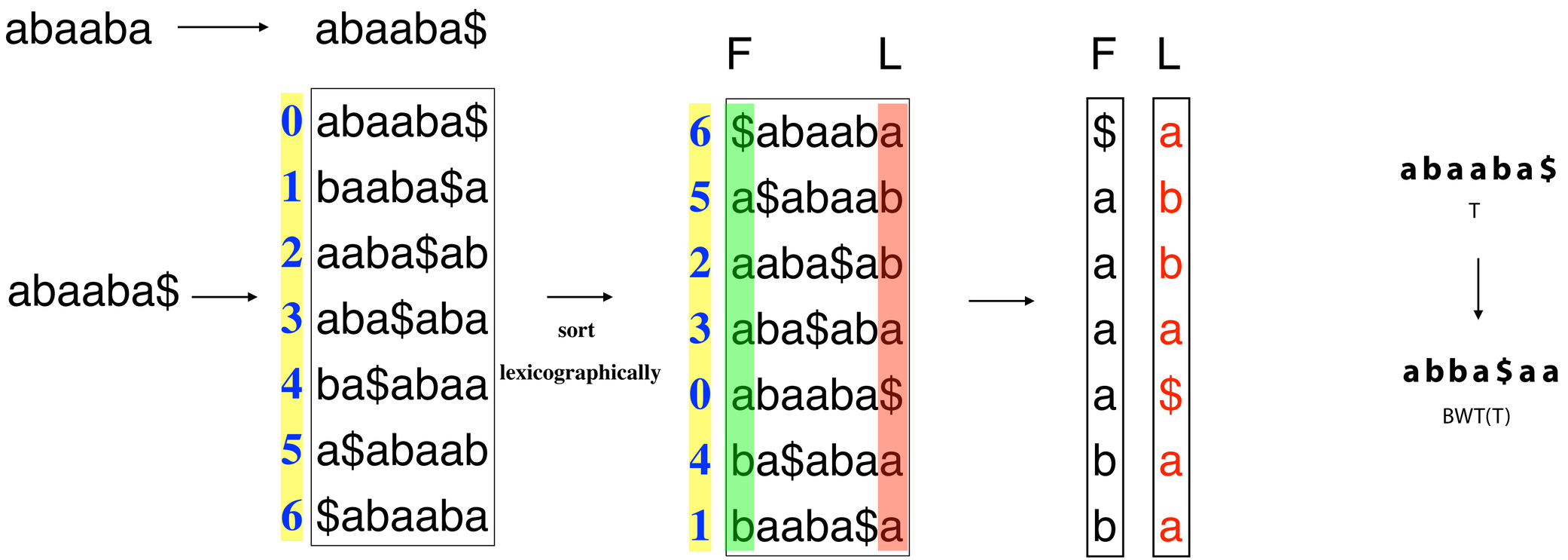
(3) The profile is compared to the protein database, again seeking local alignments. After a few minor modifications, the BLAST algorithm can be used for this directly.

(4) PSI-BLAST estimates the statistical significance of the local alignments found. Because profile substitution scores are constructed to a fixed scale, and gap scores remain independent of position, the statistical theory and parameters for gapped BLAST alignments remain applicable to profile alignments

(5) Finally, PSI-BLAST iterates, by returning to step (2), an arbitrary number of times or until convergence.

NCBI: http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-2.html

From Suffix Array to Burrows-Wheeler Transform (BWT)



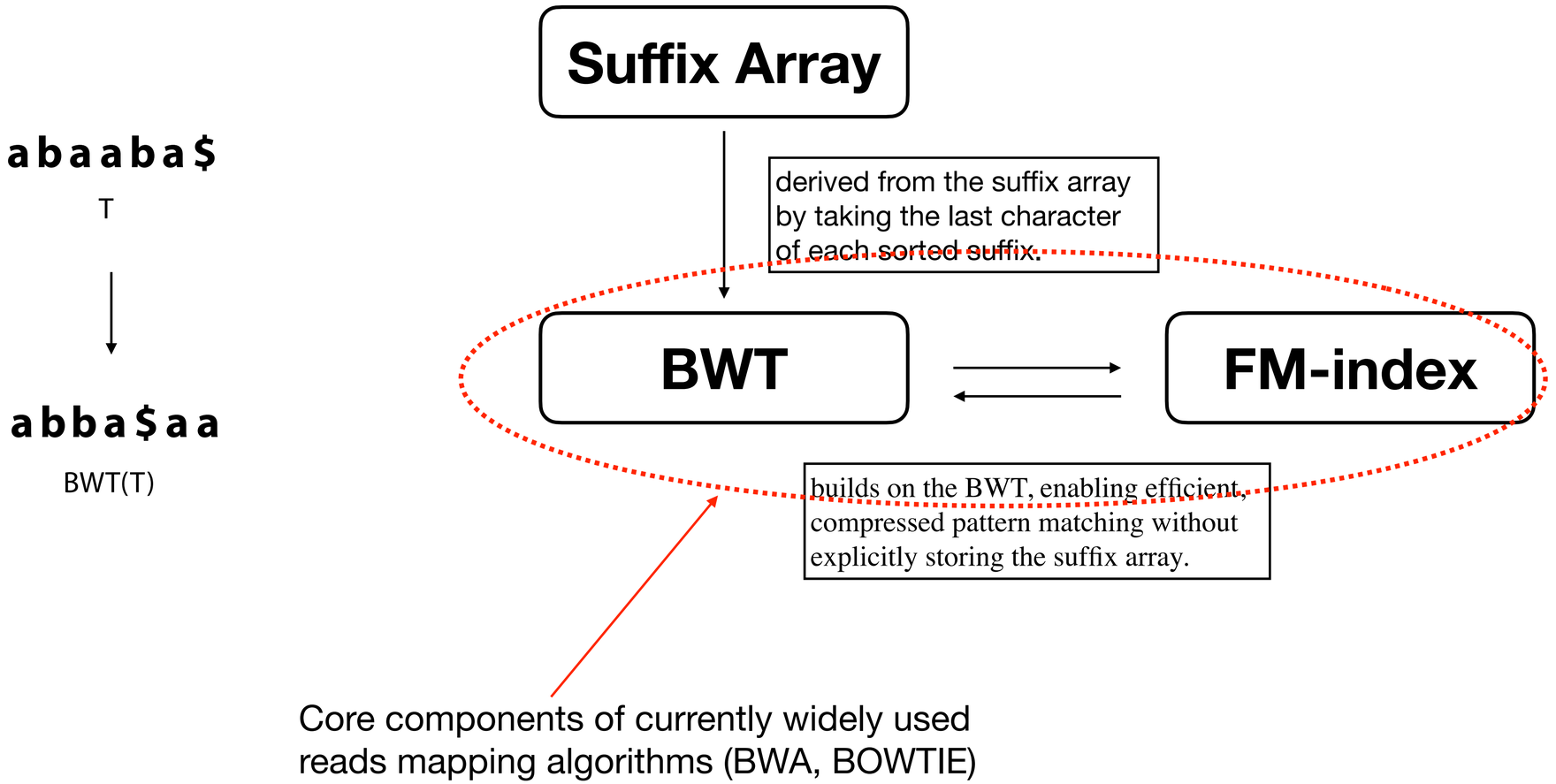
abaaba$

T

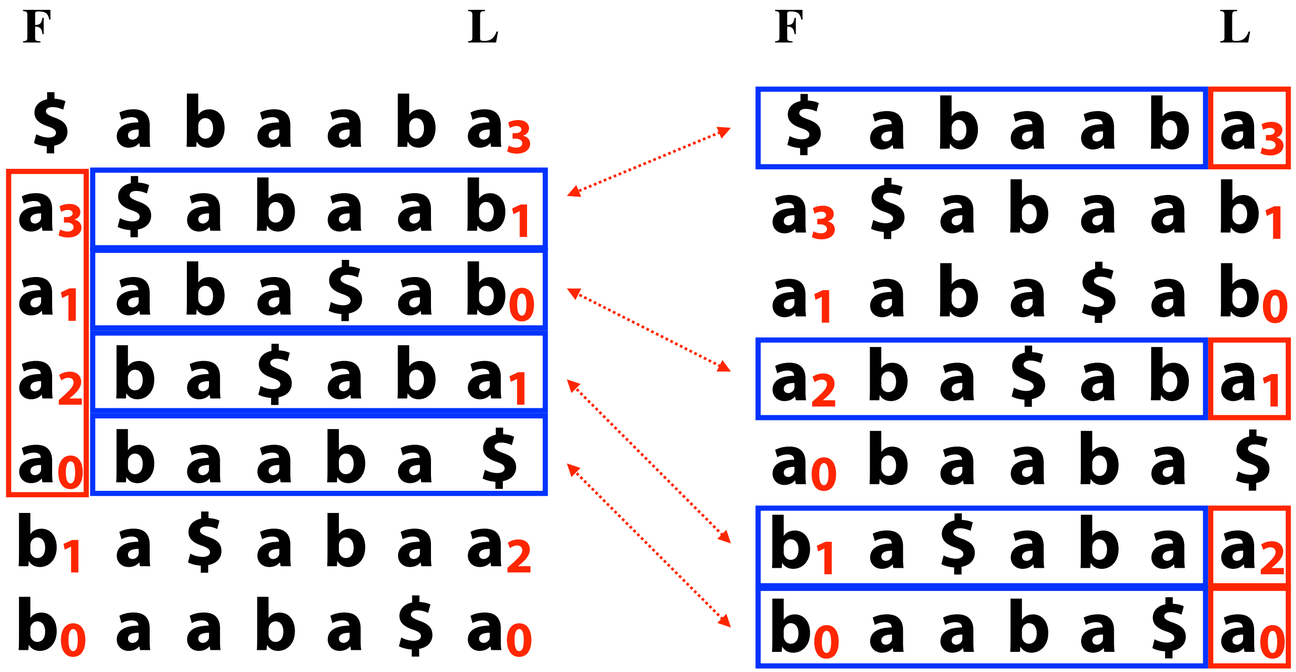
abba$aa

BWT(T)

From Suffix Array to Burrows-Wheeler Transform (BWT) and FM-index

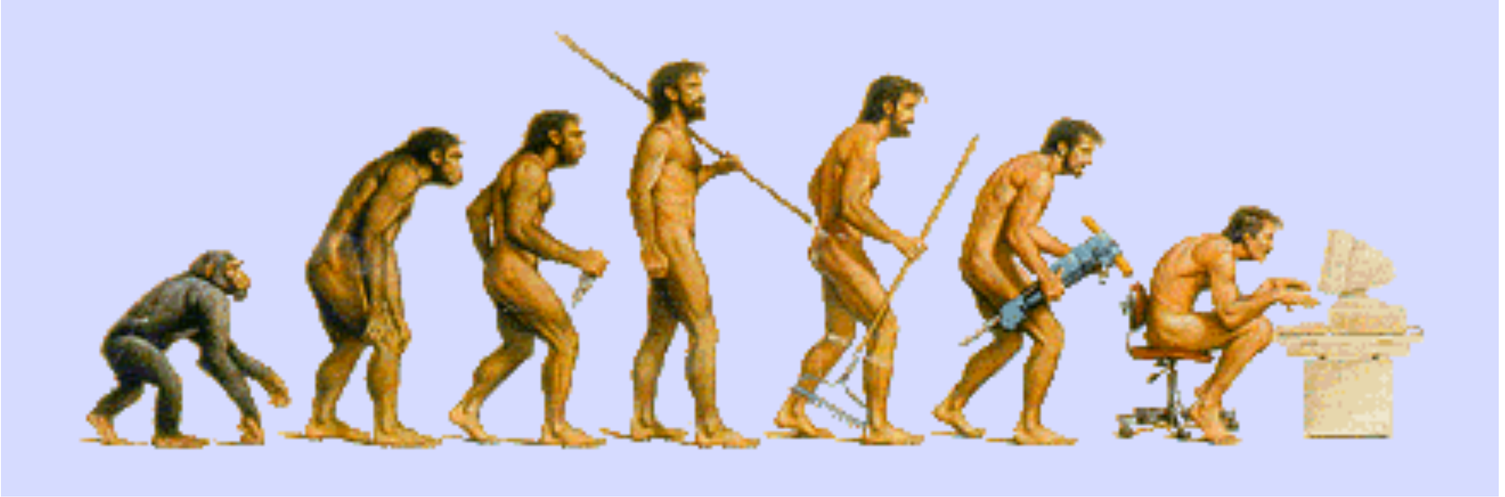


The Last-to-First (LF) mapping property in BWT



- **LF mapping** (Last-to-First mapping) is a key concept in the **Burrows-Wheeler Transform (BWT)** that enables efficient backward searching in strings. It relates the position of a character in the **last column** (BWT-transformed string) to its corresponding position in the **first column** (sorted suffixes) of the conceptual matrix used to construct the BWT.
- **LF mapping** connects the last column (BWT string) with the first column (sorted suffixes), enabling powerful backward searching capabilities in string matching algorithms

a_s are sorted by the same context in the F and the L columns



Phylogenetic Trees

- Introduction to phylogenetic trees
- Phylogenetic tree construction
  - Distance-based
  - Character-based

Phylogeny is the inference of evolutionary relationships

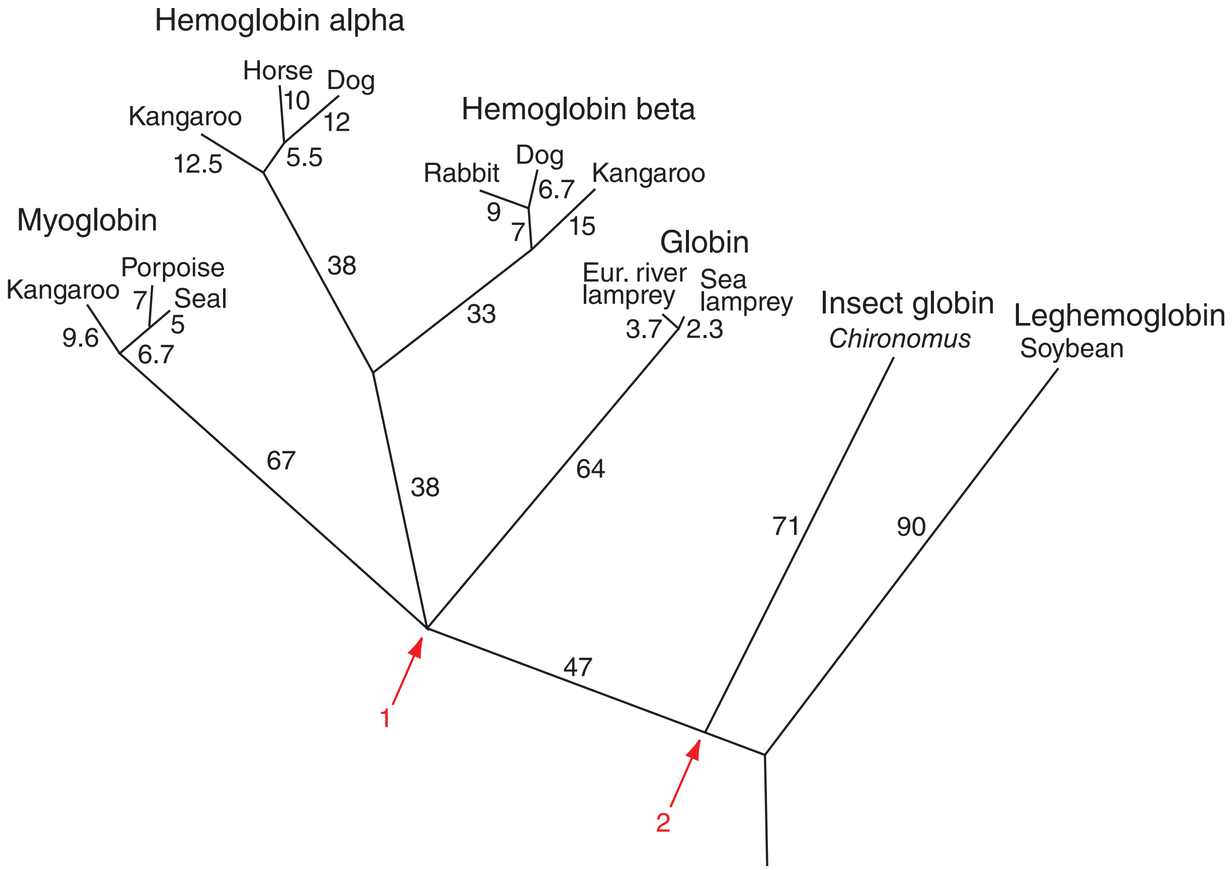


FIGURE 7.1 In the 1960s, several groups performed pioneering studies of globin phylogeny. This tree is modified from Dayhoff et al. (1972) who used maximum parsimony analysis to infer the relationships and history of 13 globins. The observed percent difference between sequences was corrected using the data on PAM matrices in **Table 3.3**. Arrow 1 indicates a node corresponding to the last common ancestor of the group of vertebrate globins, while arrow 2 indicates the ancestor of the insect and vertebrate globins (see text for details).

Source: Dayhoff et al. (1972). Reproduced with permission from National Biomedical Research Foundation.

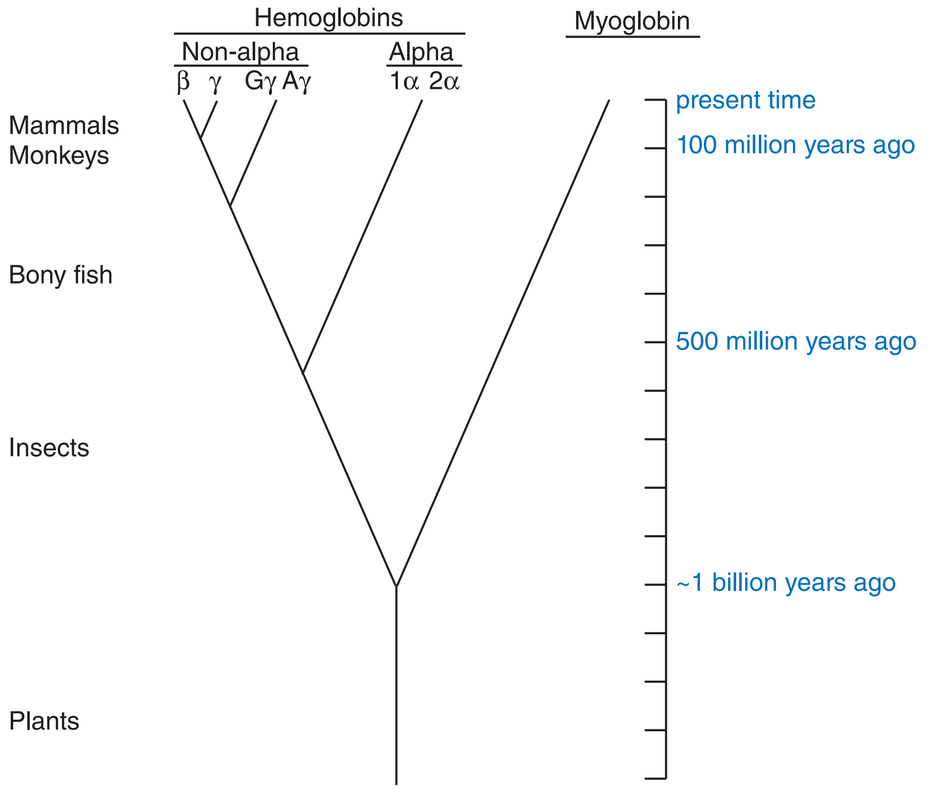


FIGURE 7.2 Dayhoff et al. (1972) summarized the relationship of the globin subfamilies in the context of evolutionary time. The dates of speciation events were inferred from fossil-based studies.

Source: Dayhoff et al. (1972). Reproduced with permission from National Biomedical Research Foundation.

Specie tree vs. gene/protein trees

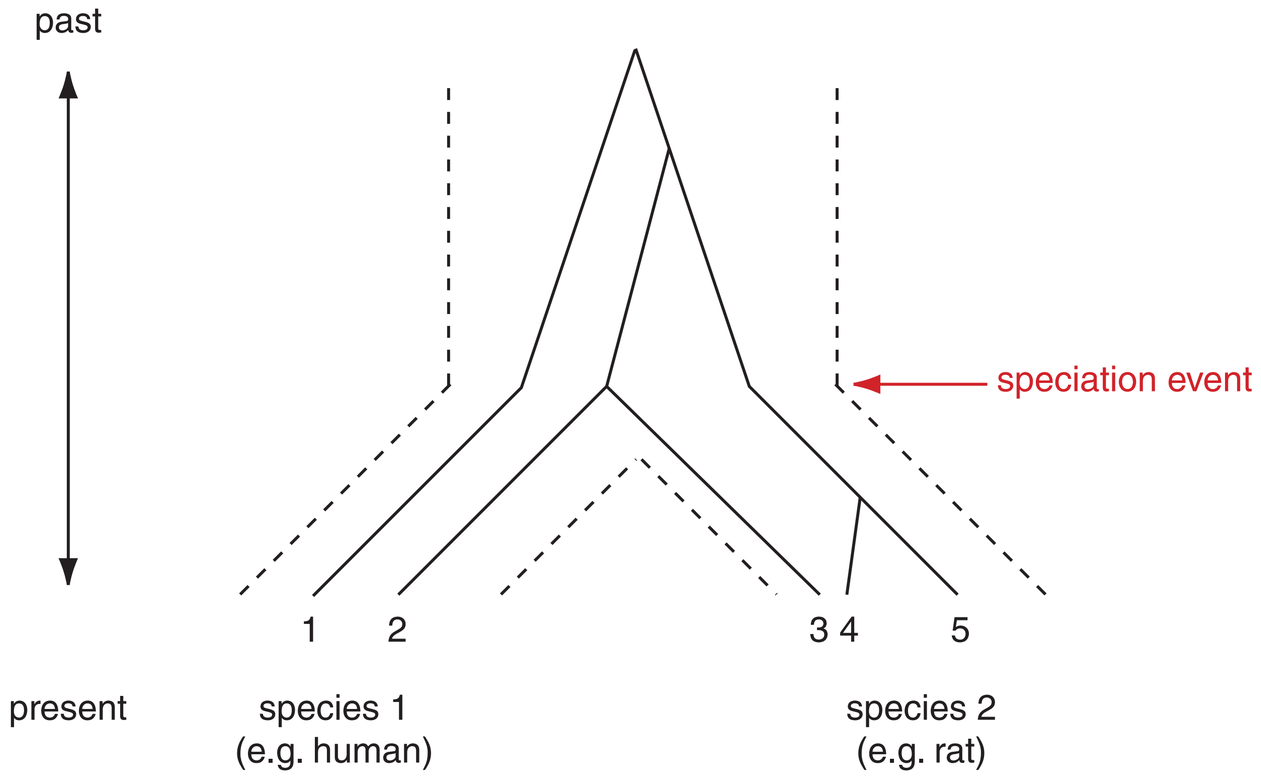
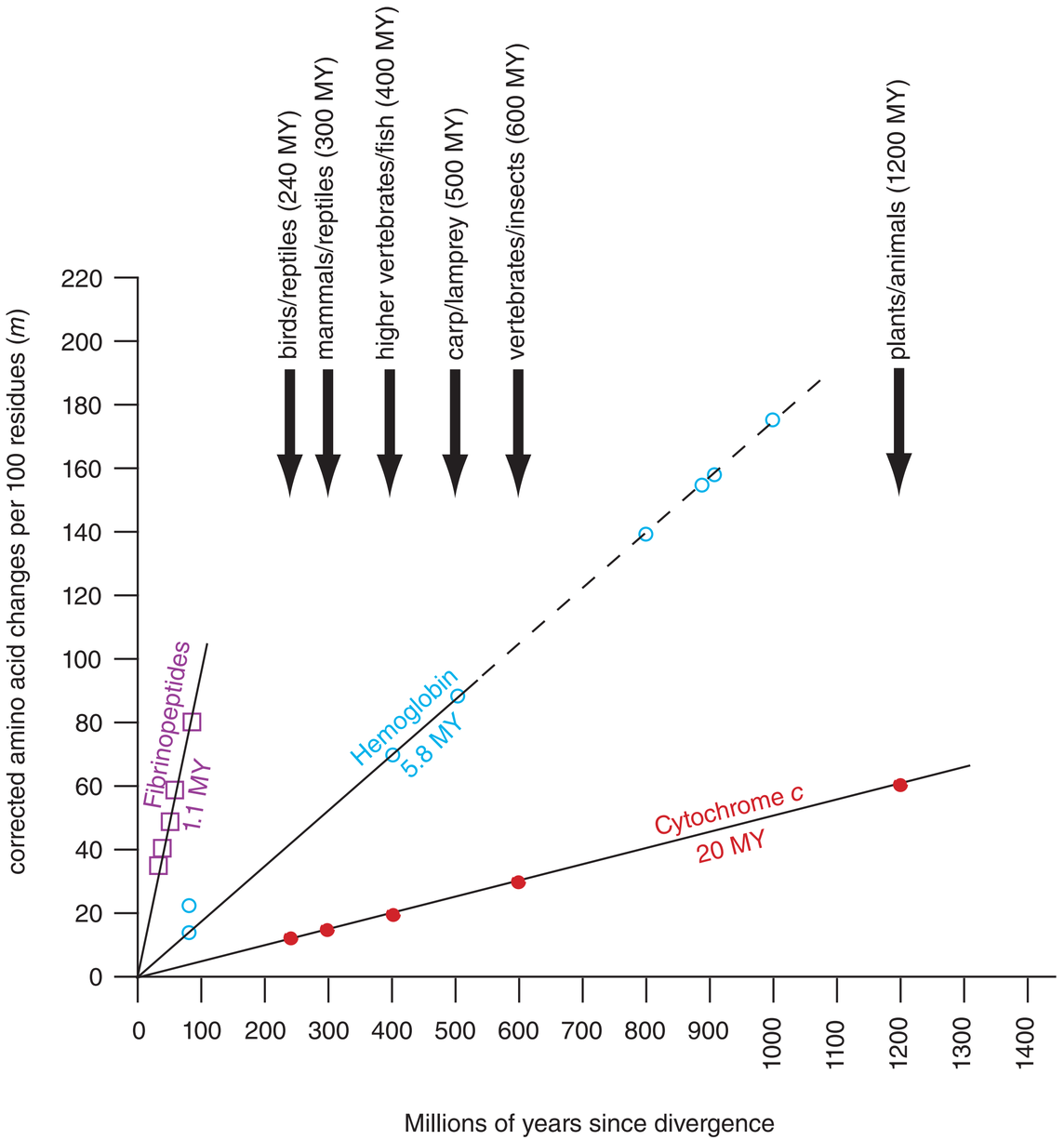


FIGURE 7.13 A species tree and a protein (or gene) tree can have a complex relationship. A speciation event, such as the divergence of the lineage that generated modern humans and rodents, can be dated to a specific time (e.g., 90 MYA). When speciation occurs, the species become reproductively isolated from one another. This event is represented by dotted lines (see horizontal arrow). Phylogenetic analysis of a specific group of homologous proteins is complicated by the fact that a gene duplication could have preceded or followed the speciation event. In essentially all phylogenetic analyses, the extant proteins (OTUs) are sequences from organisms that are alive today. It is necessary to reconstruct the history of the protein family as well as the history of each species. In the above example, there are two human paralogs and three rat paralogs. Proteins 1 and 5 diverged at a time that greatly predates the divergence of the two species. Proteins 2 and 3 diverged at a time that matches the date of species divergence. Proteins 4 and 5 diverged recently, after the time of species divergence. It is possible to reconstruct both species trees and protein (or gene) trees. Adapted from Graur and Li (2000), based upon Nei (1987). Reproduced with permission from Sinauer Associates and Columbia University Press.

- A species tree depicts the evolutionary relationships between different species, focusing on speciation events and how species have diverged from common ancestors.
- A gene or protein tree illustrates the evolutionary history of specific genes or proteins, showing gene duplications, losses, and divergences within or between species.
- While species trees represent broad evolutionary patterns, gene trees provide detailed insights into the evolution of individual genes. The two types of trees are complementary, with gene trees often used to infer or refine species relationships and vice versa.

The Molecular Clock hypothesis



- The molecular clock hypothesis suggests that genetic mutations accumulate at a relatively constant rate over time for a given protein, though rates can vary between different proteins due to functional constraints imposed by natural selection.
- These differences reflect the importance of certain proteins, with more critical ones evolving more slowly.
- By comparing the rate of genetic change in protein sequences, scientists can estimate the time of divergence between species. However, mutation rates can be affected by various factors, and calibration with external data is often needed to make accurate time estimates.

From sequences to a phylogenetic tree — Basic Assumptions on Evolution

1. **Common Ancestry**: The sequences being compared share a common ancestor, meaning they are homologous (derived from the same ancestral sequence).

2. **Mutations Accumulate Over Time**: Evolutionary changes (mutations) accumulate in sequences over time, and the number of differences between sequences correlates with the evolutionary distance between them.

3. **Constant or Predictable Mutation Rates**: A molecular clock-like assumption may be applied, suggesting that mutations occur at a relatively constant rate over time. This allows for estimating divergence times, though rates can vary across genes and species.

4. **Independent Evolution**: Once species diverge, the sequences evolve independently of one another, with mutations occurring separately in each lineage.

5. **No Recombination**: It is generally assumed that the sequences used are not affected by recombination, which could mix genetic material from different sources and confound evolutionary relationships.

6. **Selective Pressures are Similar**: It is often assumed that sequences experience similar selective pressures, although relaxed or varying selective pressures can be accounted for in certain tree-building methods (like maximum likelihood).

Phylogenetic trees

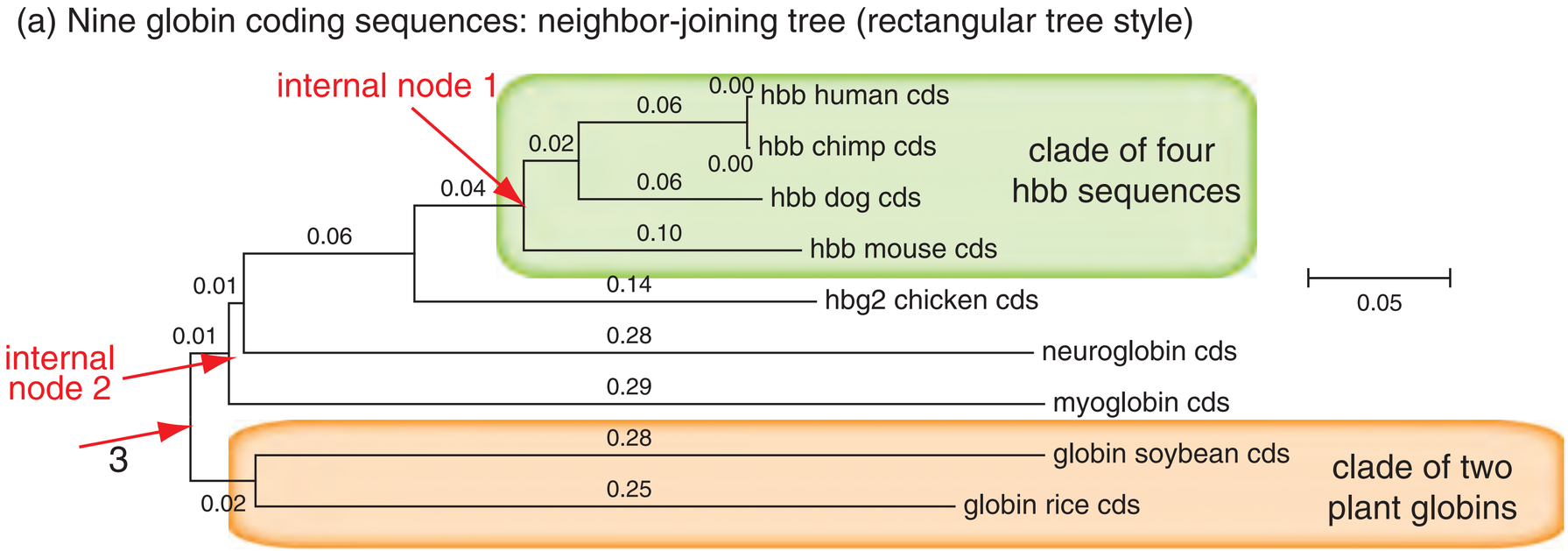


Figure **7.8** Phylogenetic trees contain nodes and branches, and are defined by the branch lengths and topology.

- The topology of a tree defines the relationships of the proteins (or other objects) that are represented in the tree.
- The branch lengths sometimes (but not always) reflect the degree of relatedness of the objects in the tree.

Phylogenetic trees

(b) Nine globin coding sequences: neighbor-joining tree ("topology only" tree style)

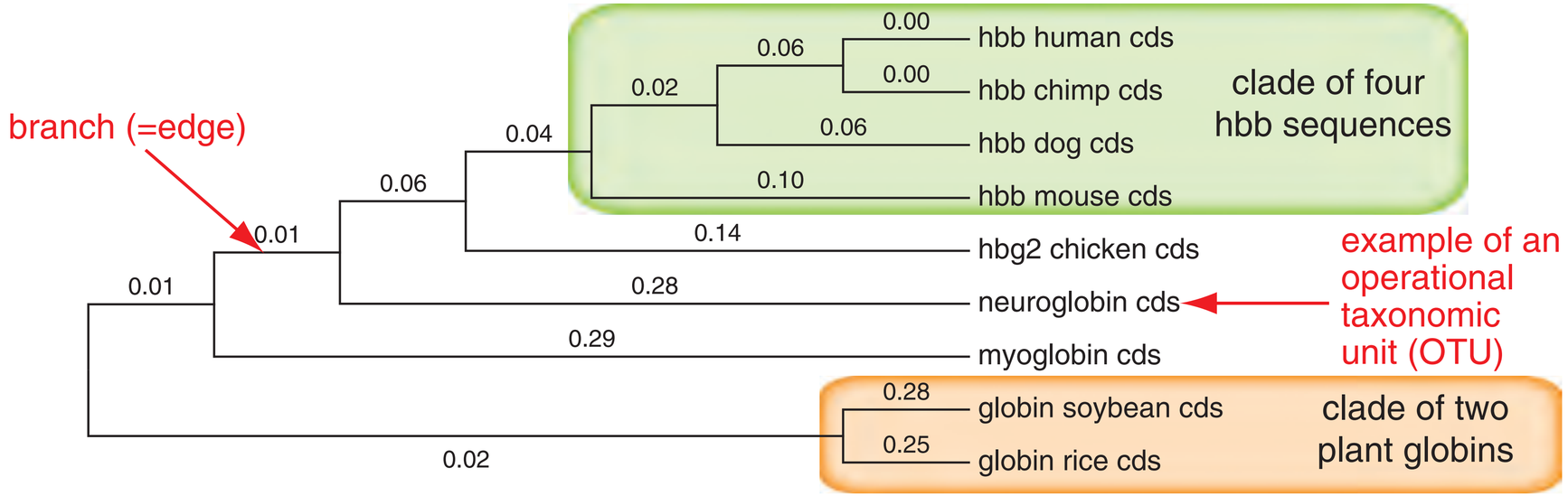


Figure **7.8** Phylogenetic trees contain nodes and branches, and are defined by the branch lengths and topology.

Phylogenetic trees

(c) Nine globin coding sequences: UPGMA tree

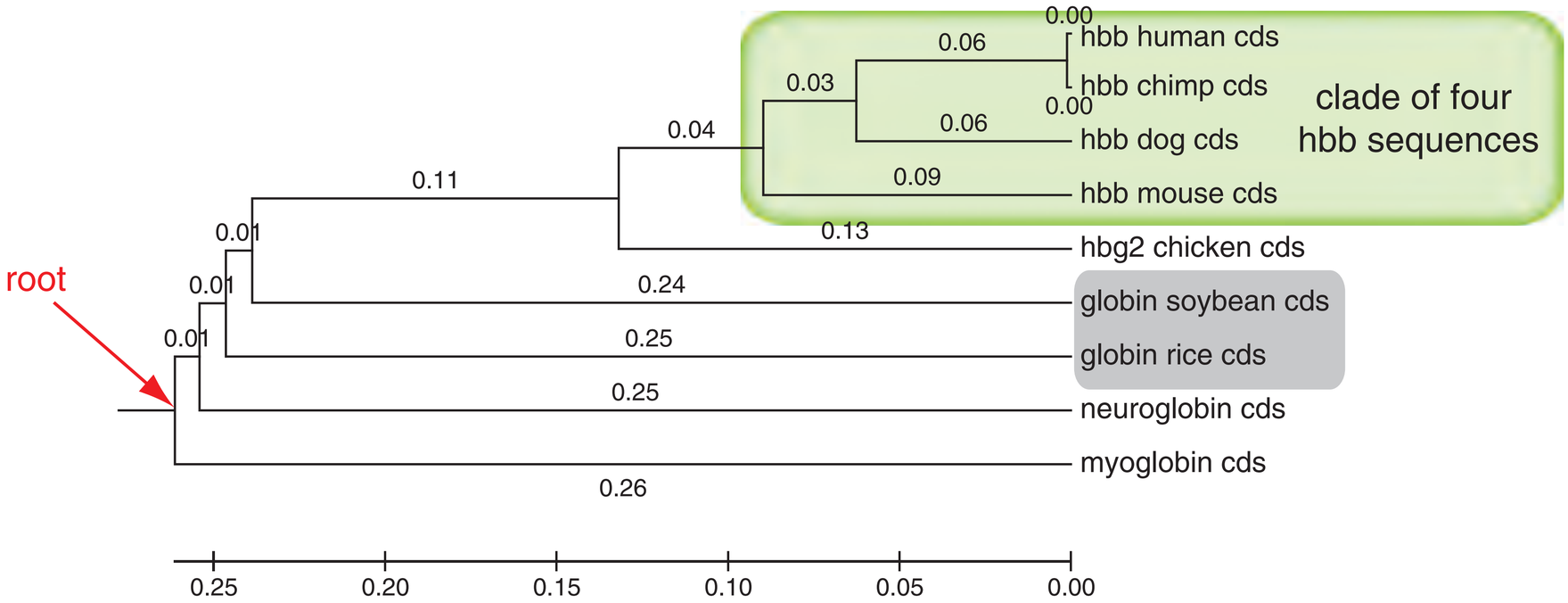


Figure **7.8** Phylogenetic trees contain nodes and branches, and are defined by the branch lengths and topology.

Phylogenetic trees

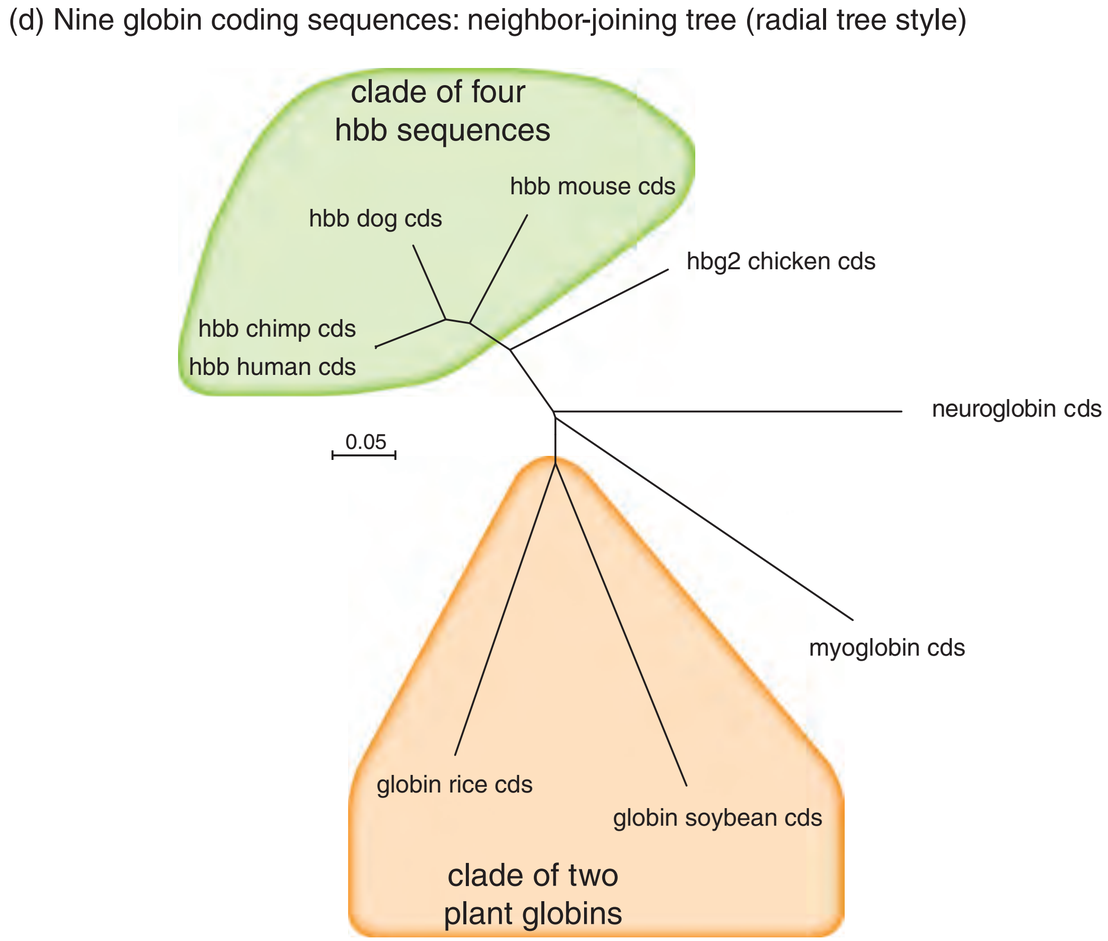
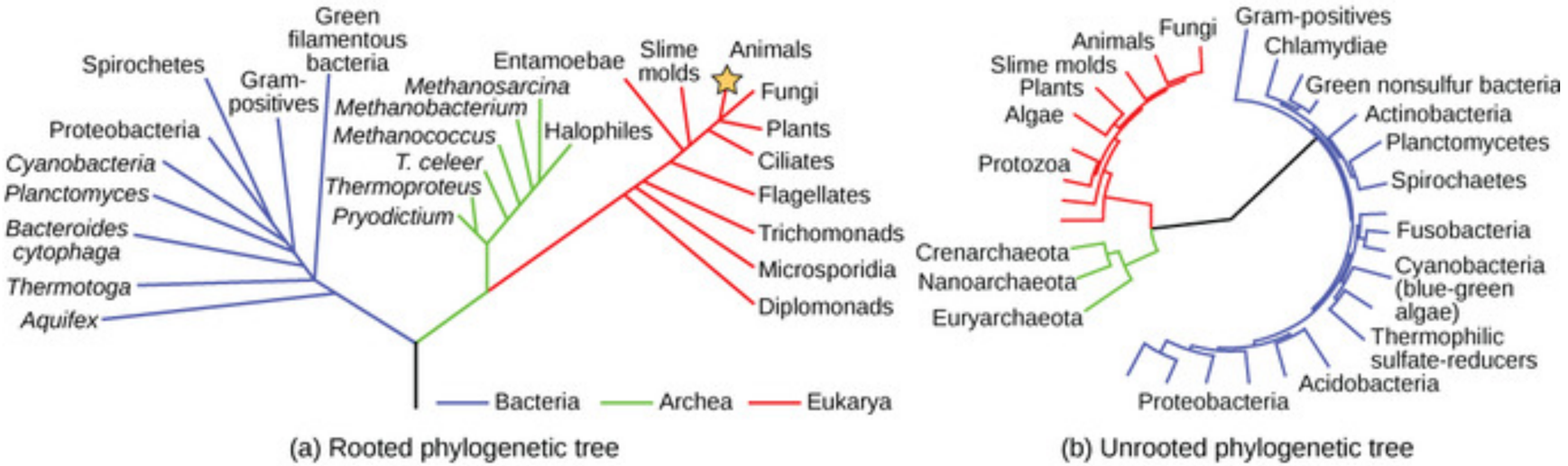


Figure **7.8** Phylogenetic trees contain nodes and branches, and are defined by the branch lengths and topology.

Rooted versus unrooted trees



(a) Rooted phylogenetic tree

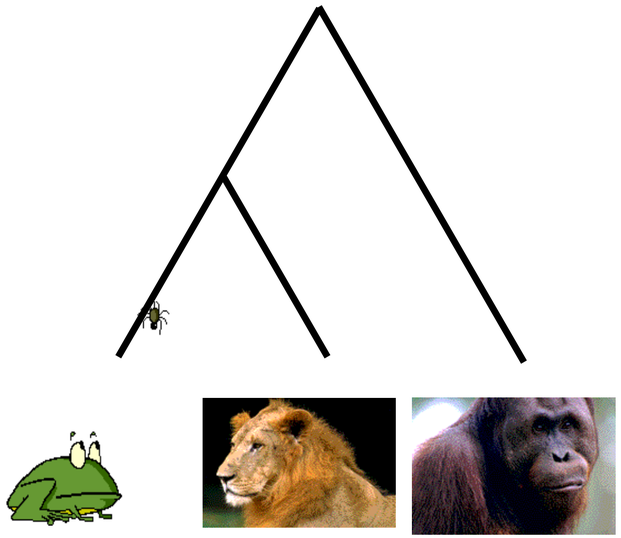
(b) Unrooted phylogenetic tree

A rooted phylogenetic tree is a type of phylogenetic tree that describes the ancestry of a group of organisms. Importantly, it is a directed tree, starting from a unique node known as the recent common ancestor. Basically, the roots of the phylogenetic tree describe this recent common ancestor.

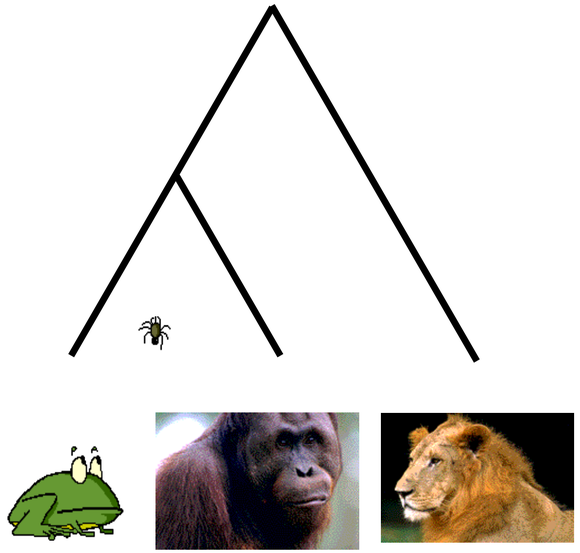
The unrooted phylogenetic tree is a type of phylogenetic tree that only describes the relatedness of a group of organisms. Importantly, the leaf nodes of this type of phylogenetic tree only show relatedness, not the ancestry. Hence, it does not start with the recent common ancestor and does not contain a root.

Rooted versus unrooted trees

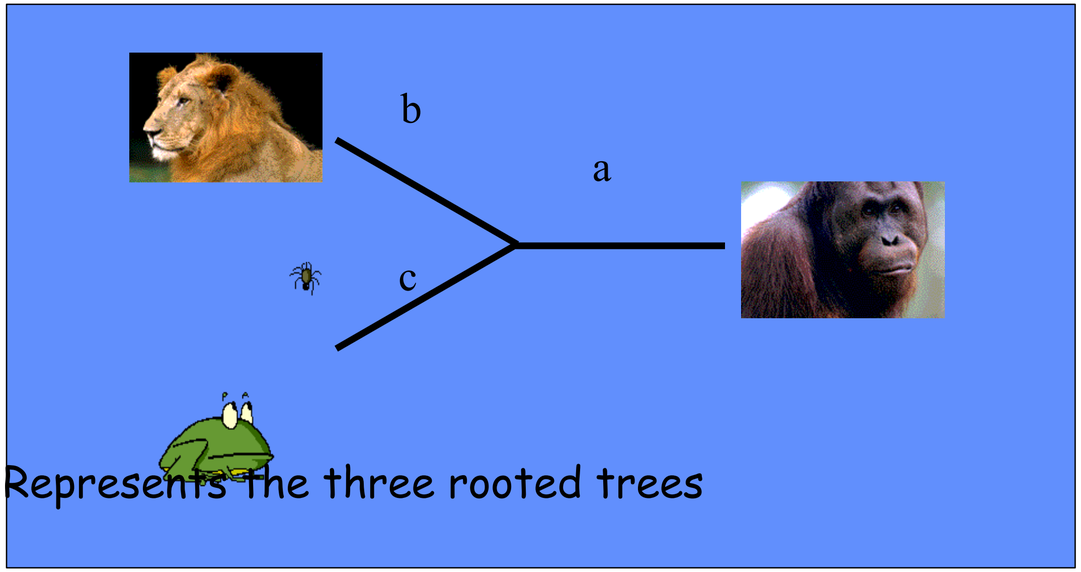
Tree A

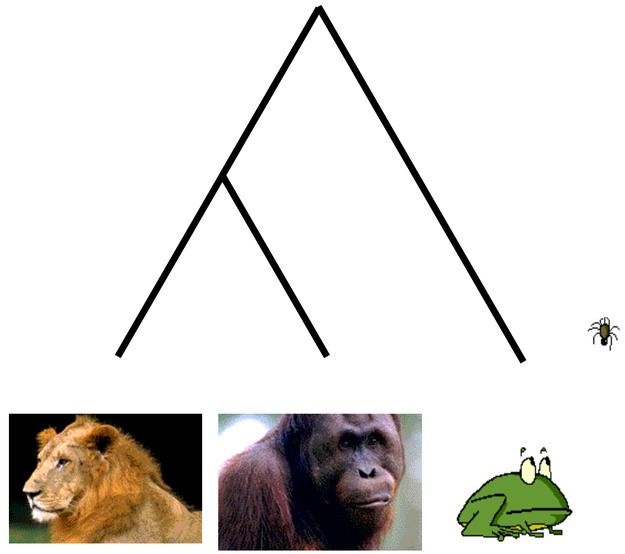


Tree B



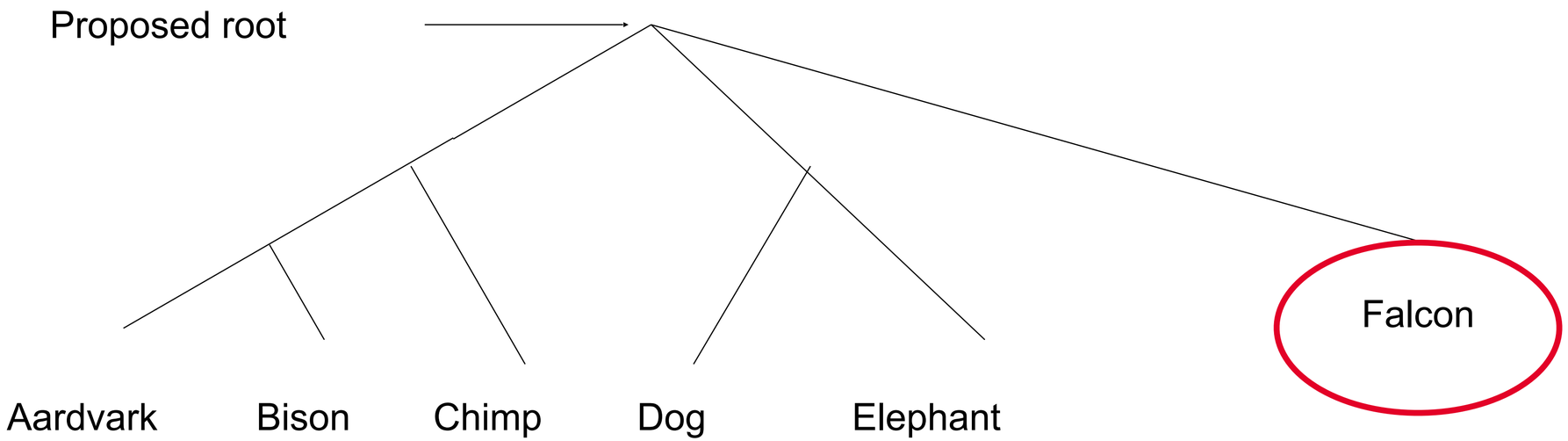
Tree C





Positioning Roots in Unrooted Trees

- The position of the root can be estimated by introducing an **outgroup**: “A species known to be more distantly related to remaining species than they are to each other.”
- The point where the outgroup joins the rest of the tree is best candidate for root position.



Methods of Tree Construction

- **Distance-Based Methods:**
  - **UPGMA (Unweighted Pair Group Method with Arithmetic Mean)**: Similar to NJ but assumes a constant rate of evolution (molecular clock). It builds the tree hierarchically, clustering pairs based on their distances.
  - **Neighbor-Joining (NJ)**: A method that constructs trees based on pairwise genetic distances, allowing for variable rates of evolution. It starts with a distance matrix and progressively merges the closest taxa.
- **Character-Based Methods:**
  - **Maximum Parsimony (MP)**: This method seeks the tree that requires the fewest evolutionary changes (mutations) to explain the observed data. It analyzes character states to minimize the total number of changes.
  - **Maximum Likelihood (ML)**: A statistical approach that evaluates the likelihood of different tree topologies given a specific model of evolution. It estimates the parameters that best explain the observed data.
- **Bayesian Methods:**
  - **Bayesian Inference**: Similar to ML but incorporates prior information and computes a posterior distribution for tree topologies. This method provides not only a single tree estimate but also uncertainty estimates for branch lengths and relationships.

Multiple sequence alignments

```
V1          PEEMSVTS-LDLTGGLPEATTPESEEAFTLPLLNDPEPK-PSLEPVKNISNMELKAEPFD
V2          PEEMSVAS-LDLTGGLPEASTPESEEAFTLPLLNDPEPK-PSLEPVKSISNVELKAEPFD
V3          SEELAAATALDLG----APSPAAAEEAFALPLMTEAPPAVPPKEPSG--SGLELKAEPFD
V4          PGPGPLAEVRDLPG-----STSAKEDGFGWLLPPPPPPP-----------------LPFQ
```

Distance based

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–

(.17 means 17 % identical)

Character based

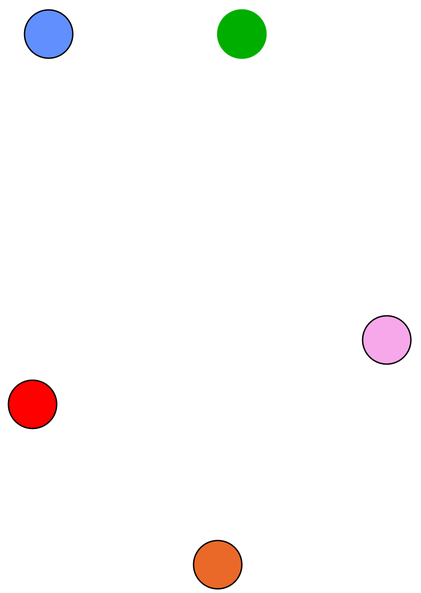
```
T
T
P
T
```

- **Parsimony**
- **Maximum likelihood**

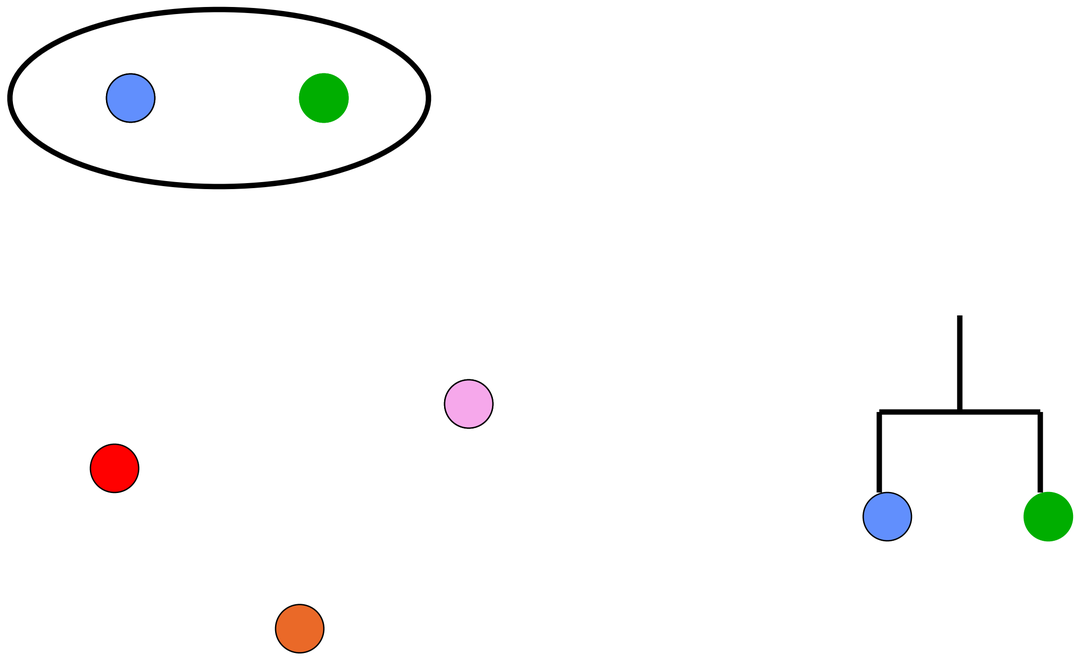
Unweighted Pair Group Method using Arithmetic Averages (UPGMA)

- **UPGMA (Unweighted Pair Group Method with Arithmetic Mean)** is a hierarchical clustering method used to construct phylogenetic trees based on genetic distance data. It assumes a constant rate of evolution (molecular clock), meaning it treats all lineages as evolving at the same rate.
  - It works by clustering the sequences, at each stage amalgamating two clusters and, at the same time, creating a new node on the tree.
  - The resulting tree is rooted and assumes equal branch lengths, which means all terminal nodes (leaves) are equidistant from the root.

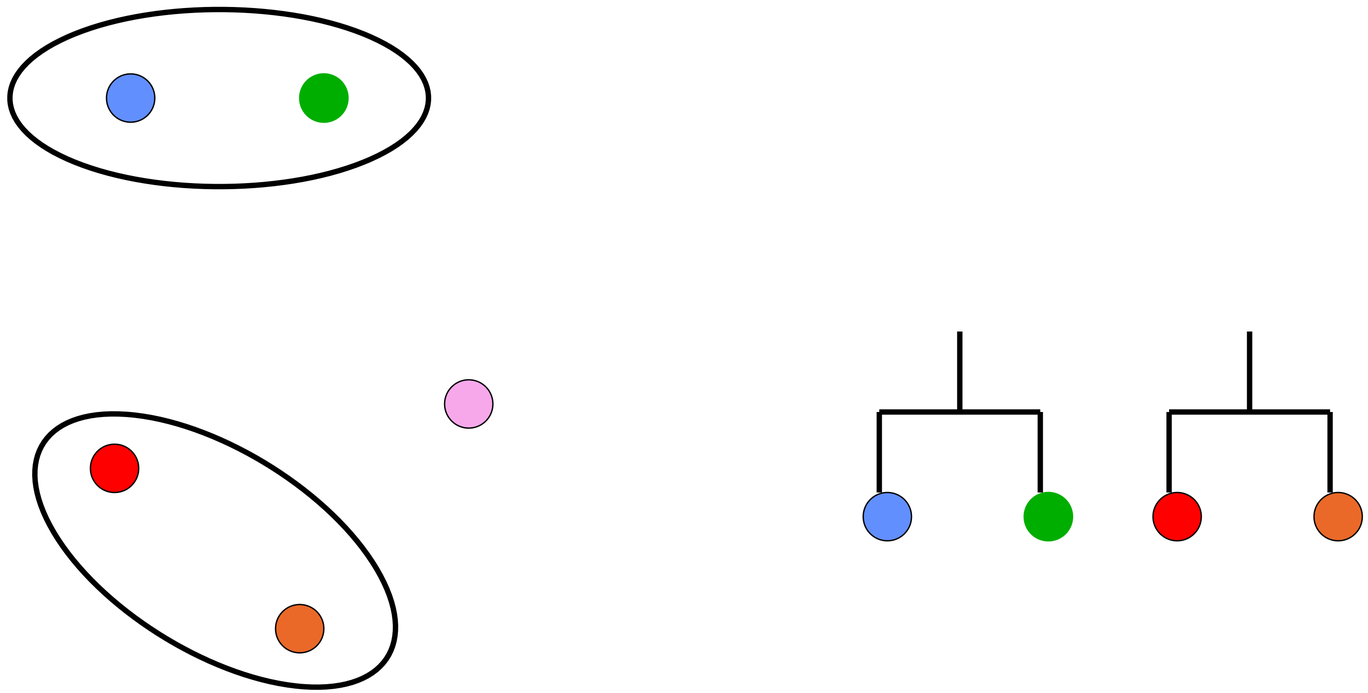
An example showing how UPGMA produces a rooted phylogenetic tree



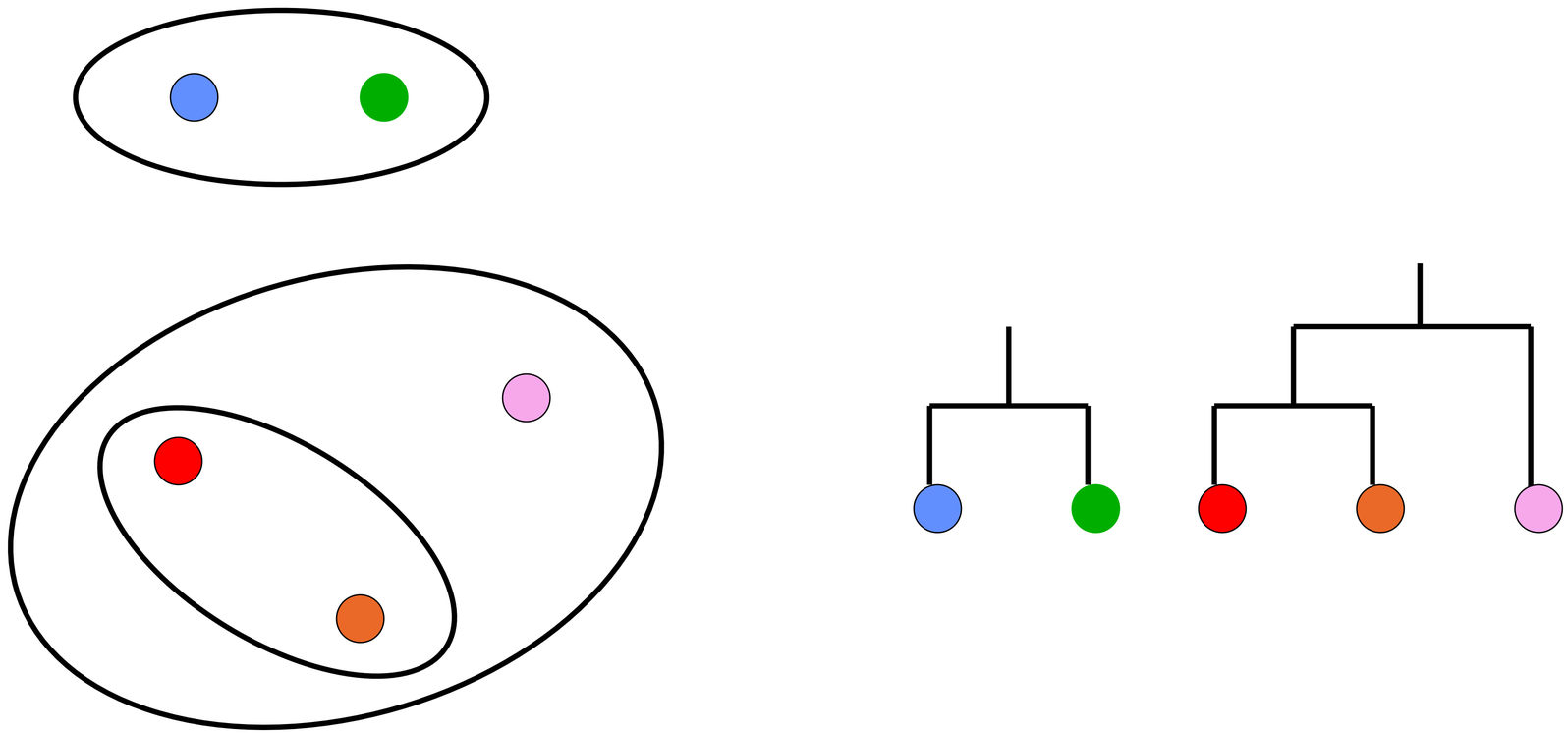
An example showing how UPGMA produces a rooted phylogenetic tree



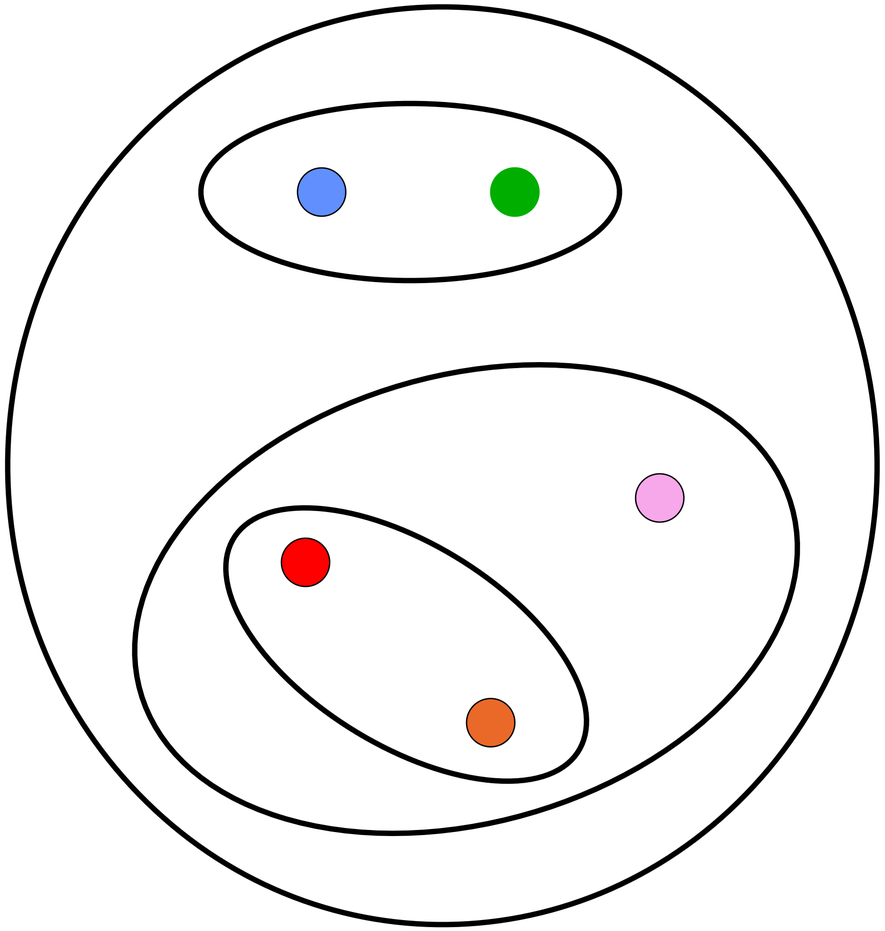
An example showing how UPGMA produces a rooted phylogenetic tree

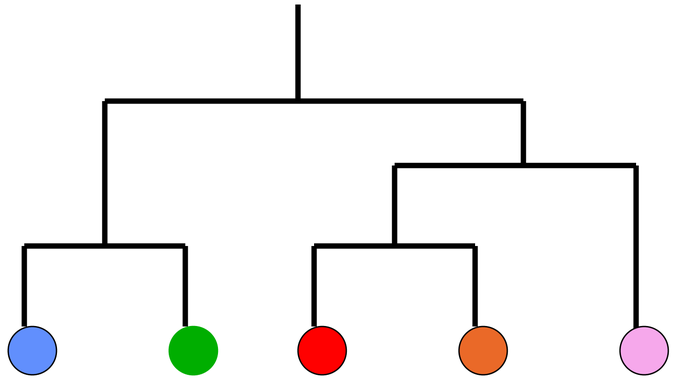


An example showing how UPGMA produces a rooted phylogenetic tree



An example showing how UPGMA produces a rooted phylogenetic tree





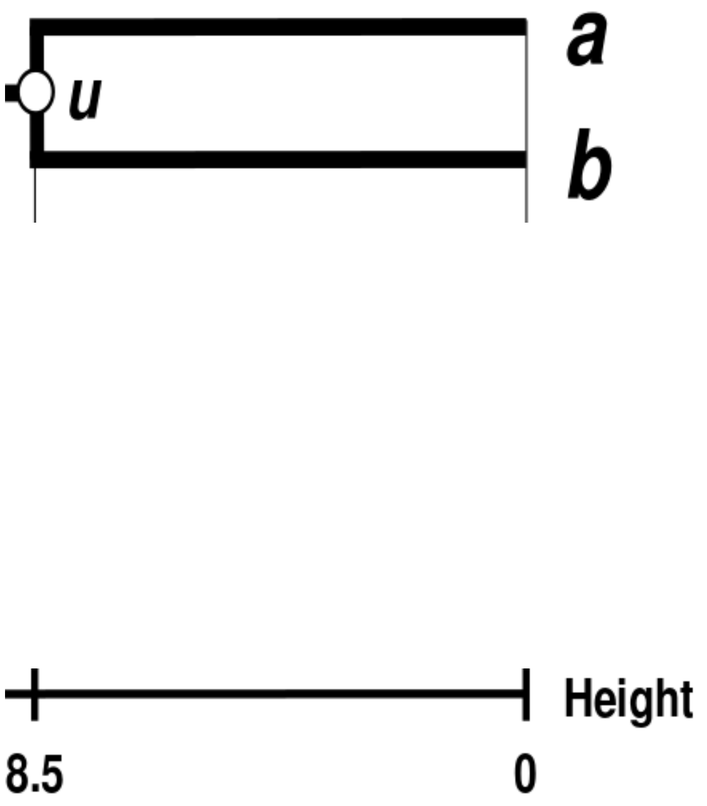
	a	b	c	d	e
a	0	17	21	31	23
b	17	0	30	34	21
c	21	30	0	28	39
d	31	34	28	0	43
e	23	21	39	43	0

- **First clustering**

$D_1(a, b) = 17$ is the smallest value of D_1, so we join elements a and b.

- **First branch length estimation**

Let u denote the node to which a and b are now connected. Setting $\delta(a, u) = \delta(b, u) = D_1(a, b)/2$ $\delta(a, u) = \delta(b, u) = 17/2 = 8.5$ ensures that elements a and b are equidistant from u.


a
u
b
Height
8.5
0

How to update distance matrix?

Given a new cluster C_k formed by merging C_i and C_j, the distance to another cluster C_l is:

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

- **First distance matrix update**

$$D_2((a,b),c) = (D_1(a,c) \times 1 + D_1(b,c) \times 1)/(1+1) = (21+30)/2 = 25.5$$

$$D_2((a,b),d) = (D_1(a,d) + D_1(b,d))/2 = (31+34)/2 = 32.5$$

$$D_2((a,b),e) = (D_1(a,e) + D_1(b,e))/2 = (23+21)/2 = 22$$

	(a,b)	c	d	e
(a,b)	0	**25.5**	**32.5**	**22**
c	**25.5**	0	*28*	*39*
d	**32.5**	*28*	0	*43*
e	**22**	*39*	*43*	0

- **Second clustering**

Here, $D_2((a,b),e) = 22$ is the smallest value of D_2, so we join cluster (a,b) and element e.

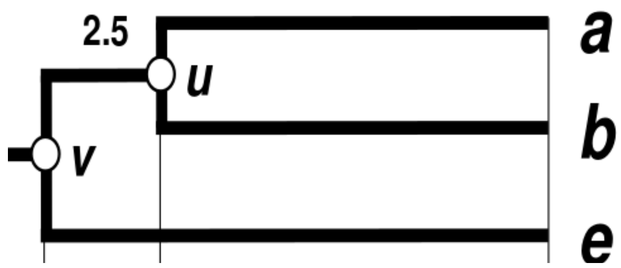
- **Second branch length estimation**

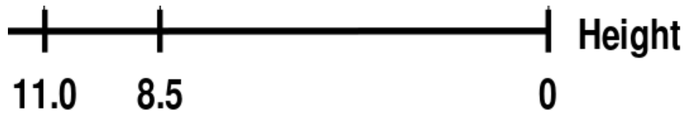
Let v denote the node to which (a, b) and e are now connected. Because of the ultrametricity constraint, the branches joining a or b to v, and e to v are equal and have the following length:

$$\delta(a, v) = \delta(b, v) = \delta(e, v) = 22/2 = 11$$

We deduce the missing branch length:

$$\delta(u, v) = \delta(e, v) - \delta(a, u) = \delta(e, v) - \delta(b, u) = 11 - 8.5 = 2.5$$





- **Second distance matrix update**

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

$$D_3(((a,b),e),c) = (D_2((a,b),c) \times 2 + D_2(e,c) \times 1)/(2+1) = (25.5 \times 2 + 39 \times 1)/3 = 30$$

$$D_3(((a,b),e),d) = (D_2((a,b),d) \times 2 + D_2(e,d) \times 1)/(2+1) = (32.5 \times 2 + 43 \times 1)/3 = 36$$

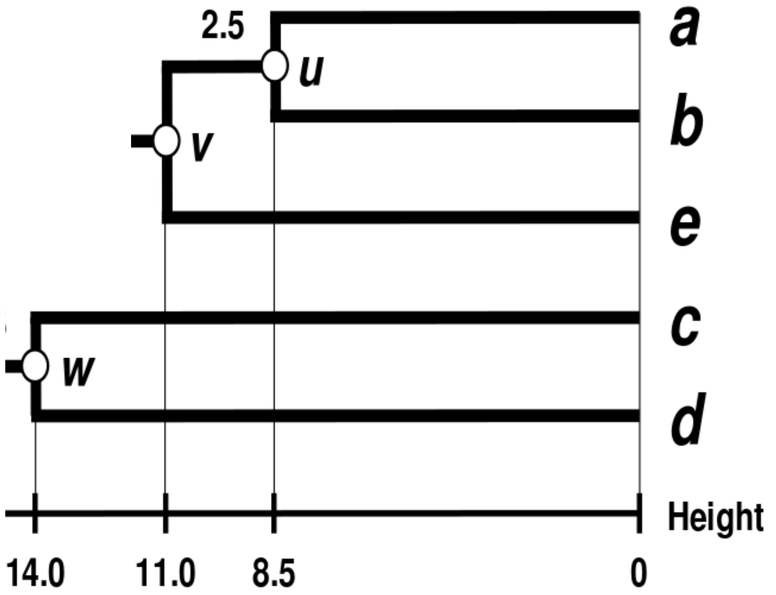
	((a,b),e)	**c**	**d**
((a,b),e)	0	**30**	**36**
c	**30**	0	*28*
d	**36**	*28*	0

- **Third clustering**

Here, $D_3(c, d) = 28$ is the smallest value of D_3, so we join elements c and d.

- **Third branch length estimation**

Let w denote the node to which c and d are now connected. The branches joining c and d to w then have lengths $\delta(c, w) = \delta(d, w) = 28/2 = 14$



- **Third distance matrix update**

There is a single entry to update, keeping in mind that the two elements c and d each have a contribution of 1 in the average computation:

$$D_4((c,d),((a,b),e)) = (D_3(c,((a,b),e)) \times 1 + D_3(d,((a,b),e)) \times 1)/(1+1) = (30 \times 1 + 36 \times 1)/2 = 33$$

The final D_4 matrix is:

	((a,b),e)	(c,d)
((a,b),e)	0	**33**
(c,d)	**33**	0

So we join clusters $((a,b),e)$ and (c,d).

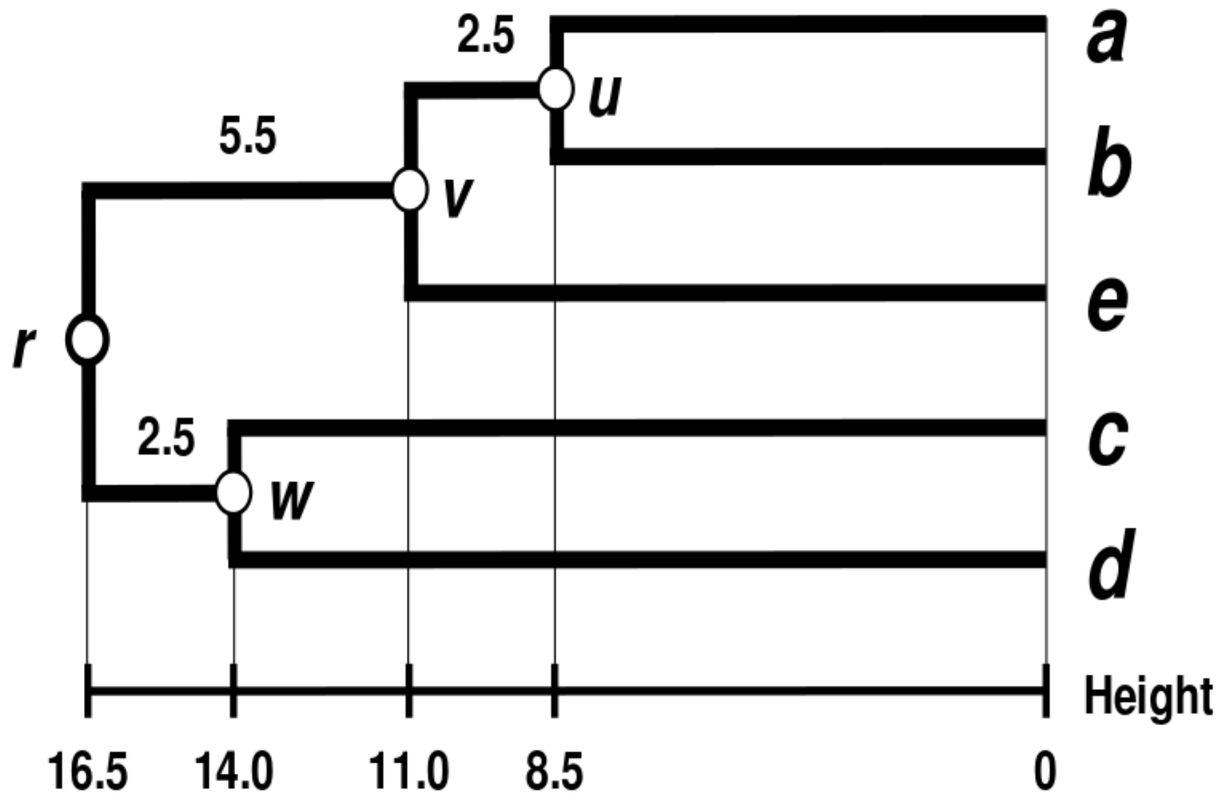
Let r denote the (root) node to which $((a,b),e)$ and (c,d) are now connected. The branches joining $((a,b),e)$ and (c,d) to r then have lengths:

$$\delta(((a,b),e),r) = \delta((c,d),r) = 33/2 = 16.5$$

We deduce the two remaining branch lengths:

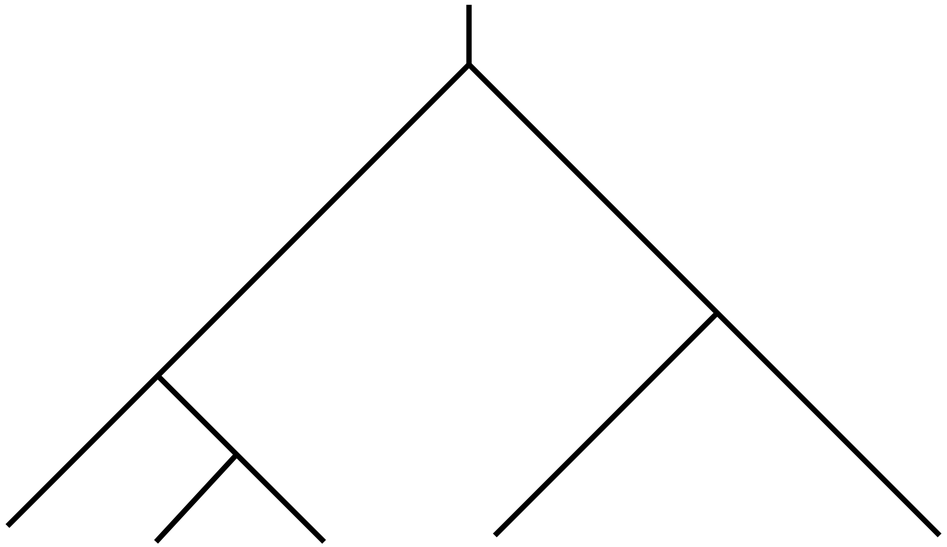
$$\delta(v,r) = \delta(((a,b),e),r) - \delta(e,v) = 16.5 - 11 = 5.5$$

$$\delta(w,r) = \delta((c,d),r) - \delta(c,w) = 16.5 - 14 = 2.5$$


2.5
u
a
5.5
v
b
e
r
2.5
w
c
d
Height
16.5
14.0
11.0
8.5
0

Molecular clock

The tree produced by UPGMA assumes a molecular clock with a constant rate.



This phylogenetic tree has all leaves in the same level. When this property holds, the phylogenetic tree is said to satisfy a molecular clock. Namely, the time from a speciation event to the formation of current species is identical for all paths (wrong assumption in reality).

Molecular Clock

If the original distance data are obtained by summing edge lengths in a tree that follows a molecular clock, UPGMA can accurately reconstruct the tree. However, if the original data do not adhere to this pattern, UPGMA may produce an incorrect tree reconstruction.

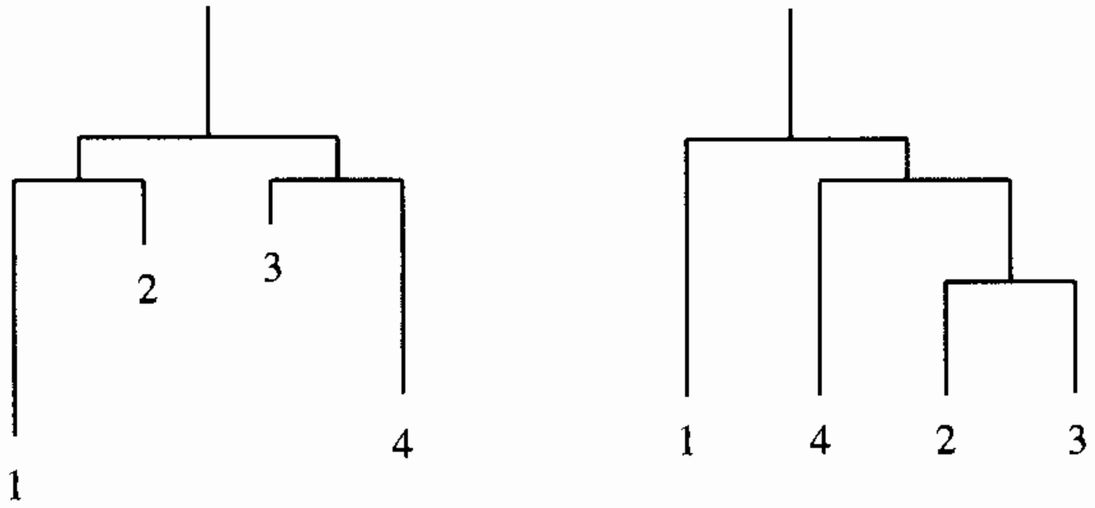


Figure 7.5 *A tree (left) that is reconstructed incorrectly by UPGMA (right).*

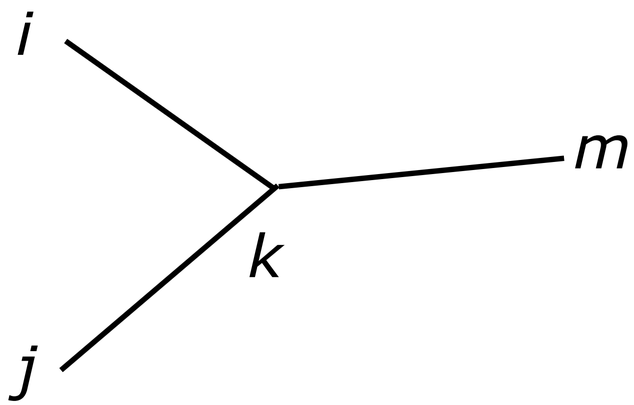
Additivity

Besides the molecular clock properties of the tree produced by UPGMA, there is another implicitly assumed property: additivity.

Additivity: Additivity means that the distance matrix used in constructing the tree accurately reflects the sum of branch lengths along the path between any two taxa. This property ensures that the total distance calculated using the tree matches the pairwise distances in the distance matrix.

- When the molecular clock assumption fails, it means that the rate of molecular evolution varies significantly across different lineages. This can lead to incorrect tree topologies if methods relying heavily on the clock assumption (like UPGMA) are used.
- If additivity still holds, it means that while the evolutionary rates may differ, the distances calculated between taxa can still represent the true branch lengths in the tree. This allows the NJ method to construct a tree that reflects the true relationships among taxa based on the distances.

Additivity



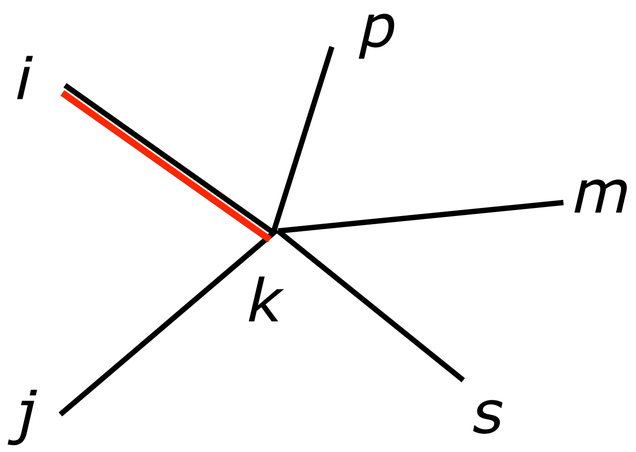
$$d_{im} = d_{ik} + d_{km}$$

$$d_{jm} = d_{jk} + d_{km}$$

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

- Additivity in the context of phylogenetics refers to the property that the distances between taxa can be represented as sums of the branch lengths in a tree, i.e., the distance between any two taxa should equal the sum of the lengths of the branches connecting them.

- When additivity holds, the distances can be directly interpreted as the lengths of paths on the tree.



$$d(i,k) = \frac{d(i,j) + d(i,m) - d(j,m)}{2}$$

$$d(i,k) = \frac{d(i,j) + d(i,p) - d(j,p)}{2}$$

$$d(i,k) = \frac{d(i,j) + d(i,s) - d(j,s)}{2}$$

$$3 \cdot d(i,k) = \frac{3 \cdot d(i,j) + (d(i,m) + d(i,p) + d(i,s)) - (d(j,m) + d(j,p) + d(j,s))}{2}$$

$$3 \cdot d(i,k) = \frac{3 \cdot d(i,j) + (d(i,m) + d(i,p) + d(i,s) + d(i,j)) - (d(j,m) + d(j,p) + d(j,s) + d(i,j))}{2}$$

$$d(i) = \sum_{i \neq m} d(i,m) \qquad d(j) = \sum_{j \neq m} d(j,m)$$

$$d(i,k) = \frac{d(i,j) + \frac{d(i) - d(j)}{n-2}}{2}$$

How can we find the neighboring leaves from the distances alone?

Saitou & Nei method: the neighboring leaves are not only ***close to each other*** (as in UPGMA) but also are ***far apart from the rest***.

$$D_{corr}(i,j) = d(i,j) - \frac{d(i) + d(j)}{n-2}$$

where: $d(i,j)$ is the raw distance between taxa i and j. $d(i)$ is the total distance from taxon i to all other taxa. $d(j)$ is the total distance from taxon j to all other taxa. n is the total number of original taxa.

Neighbor Joining Algorithm

1. **Input:**
   - A distance matrix $d(i, j)$ for all taxa.
2. **Initialize Tree:**
   - Treat each taxon as a separate leaf node in the tree.
3. **Iterative Merging:**
   - While more than two taxa remain:
     (a) **Calculate Q-Matrix:**

$$Q(i, j) = d(i, j) - \frac{d(i) + d(j)}{n - 2}$$

     where n is the number of remaining taxa. Find the pair (i, j) with the smallest $Q(i, j)$.
     (b) **Merge Taxa:**
     – Create a new cluster (ij) by merging taxa i and j.
     (c) **Update Distances:** For each remaining taxon k:

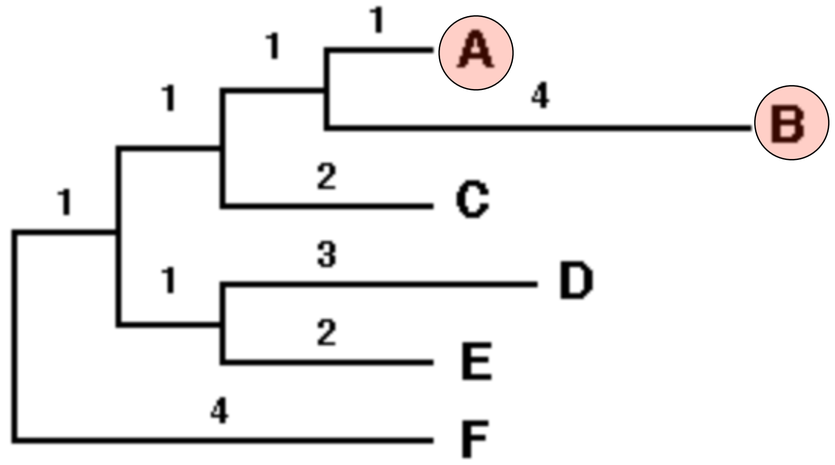
$$d((ij), k) = \frac{d(i, k) + d(j, k) - d(i, j)}{2}$$

     Remove rows and columns for i and j from the distance matrix, and add a row and column for the new cluster (ij).
4. **Final Connection:**
   - When only two taxa remain, connect them directly.
5. **Output:**
   - Construct and return the phylogenetic tree based on the merged clusters and their distances.

Distance matrix between six taxa (A - F)

	A	B	C	D	E	F
A	0	5	4	7	6	8
B	5	0	7	10	9	11
C	4	7	0	7	6	8
D	7	10	7	0	5	9
E	6	9	6	5	0	8
F	8	11	8	9	8	0

The tree constructed by neighbor joining



Step by step calculation

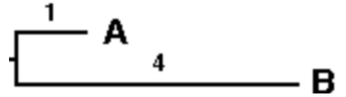
1. Calculate the total distance of each taxon

$$d(A) = 5 + 4 + 7 + 6 + 8 = 30$$
$$d(B) = 5 + 7 + 10 + 9 + 11 = 42$$
$$d(C) = 4 + 7 + 7 + 6 + 8 = 32$$
$$d(D) = 7 + 10 + 7 + 5 + 9 = 38$$
$$d(E) = 6 + 9 + 6 + 5 + 8 = 34$$
$$d(F) = 8 + 11 + 8 + 9 + 8 = 44$$

2. Calculate the corrected distances (Q-matrix)

$$\boxed{D_{corr}(i,j) = d(i,j) - \frac{d(i) + d(j)}{n-2}}$$

$$Q(A,B) = 5 - \frac{30+42}{4} = 5 - 18 = -13$$
$$Q(A,C) = 4 - \frac{30+32}{4} = 4 - 15.5 = -11.5$$
$$Q(A,D) = 7 - \frac{30+38}{4} = 7 - 17 = -10$$
$$Q(A,E) = 6 - \frac{30+34}{4} = 6 - 16 = -10$$
$$Q(A,F) = 8 - \frac{30+44}{4} = 8 - 18.5 = -10.5$$



4. Calculate the distances to internal node

$$d(A,(AB)) = \frac{d(A,B) + \frac{d(A)-d(B)}{n-2}}{2} = \frac{5 + \frac{30-42}{4}}{2} = 1$$

$$d(B,(AB)) = d(A,B) - d(A,(AB)) = 5 - 1 = 4$$

3. Identify the closest pair

	A	B	C	D	E	F
A	0	-13	-11.5	-10	-10	-10.5
B	-13	0	-11.5	-10	-10	-10.5
C	-11.5	-11.5	0	-10.5	-10.5	-11
D	-10	-10	-10.5	0	-13	-11.5
E	-10	-10	-10.5	-13	0	-11.5
F	-10.5	-10.5	-11	-11.5	-11.5	0

5. Update the distance matrix

$$d((AB),C) = \frac{d(A,C) + d(B,C) - d(A,B)}{2} = \frac{4+7-5}{2} = 3$$

$$d((AB),D) = \frac{d(A,D) + d(B,D) - d(A,B)}{2} = \frac{7+10-5}{2} = 6$$

$$d((AB),E) = \frac{d(A,E) + d(B,E) - d(A,B)}{2} = \frac{6+9-5}{2} = 5$$

$$d((AB),F) = \frac{d(A,F) + d(B,F) - d(A,B)}{2} = \frac{8+11-5}{2} = 7$$

	(AB)	C	D	E	F
(AB)	0	3	6	5	7
C	3	0	7	6	8
D	6	7	0	5	9
E	5	6	5	0	8
F	7	8	9	8	0

6. Calculate the total distances

$$d(AB) = 3+6+5+7 = 21$$
$$d(C) = 3+7+6+8 = 24$$
$$d(D) = 6+7+5+9 = 27$$
$$d(E) = 5+6+5+8 = 24$$
$$d(F) = 7+8+9+8 = 32$$

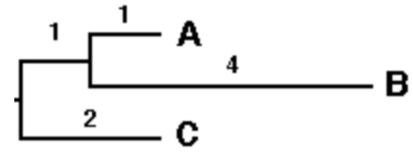
7. Calculate the corrected distance matrix Q

	(AB)	C	D	E	F
(AB)	0	-12	-10	-10	-10.67
C	-12	0	-10	-10	-10.67
D	-10	-10	0	-12	-10.67
E	-10	-10	-12	0	-10.67
F	-10.67	-10.67	-10.67	-10.67	0

8. Identify the closest pair and calculate the distance to internal node

$$d((AB),(ABC)) = \frac{d((AB),C) + \frac{d((AB))-d(C)}{n-2}}{2} = \frac{3 + \frac{21-24}{3}}{2} = 1$$

$$d(C,(ABC)) = d((AB),C) - d((AB),(ABC)) = 3 - 1 = 2$$



Updated distance matrix

	(ABC)	D	E	F
(ABC)	0	5	4	6
D	5	0	5	9
E	4	5	0	8
F	6	9	8	0

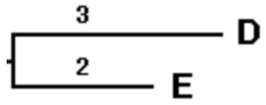
→

New corrected distance matrix

	(ABC)	D	E	F
(ABC)	0	-6.33	-6.67	-6.67
D	-6.33	0	-7	-5
E	-6.67	-7	0	-5.33
F	-6.67	-5	-5.33	0

→

New closest pair



Updated distance matrix

	(ABC)	(DE)	F
(ABC)	0	2	6
(DE)	2	0	6
F	6	6	0

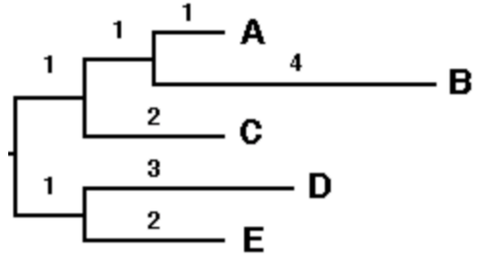
→

New corrected distance matrix

	(ABC)	(DE)	F
(ABC)	0	-6	-4
(DE)	-6	0	-4
F	-4	-4	0

→

New closest pair



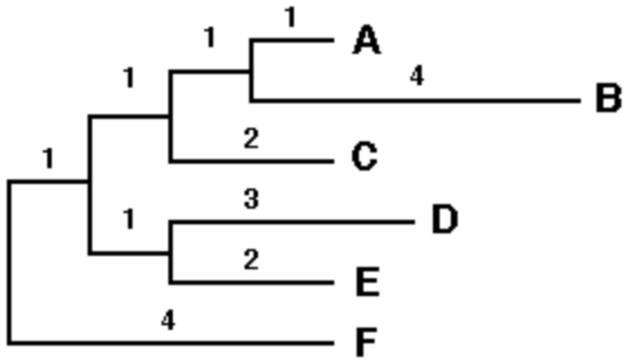


Table 1: Comparison of Neighbor Joining (NJ) and UPGMA Methods

Feature	Neighbor Joining (NJ)
Method Type	Distance-based method
Assumption	Additivity of distances
Time Complexity	$O(n^3)$
Tree Type	Unrooted tree
Sensitivity	Sensitive to distance measures
Handling of Molecular Clock	Does not require the clock
Merge Criteria	Minimizes total branch length
Statistical Support	Can be combined with bootstrapping
Feature	**UPGMA**
Method Type	Hierarchical clustering method
Assumption	Molecular clock holds
Time Complexity	$O(n^3)$
Tree Type	Rooted tree
Sensitivity	Less sensitive to distance measures
Handling of Molecular Clock	Assumes a constant rate
Merge Criteria	Average distance between clusters
Statistical Support	Does not provide inherent support

Disadvantage of distance-based methods

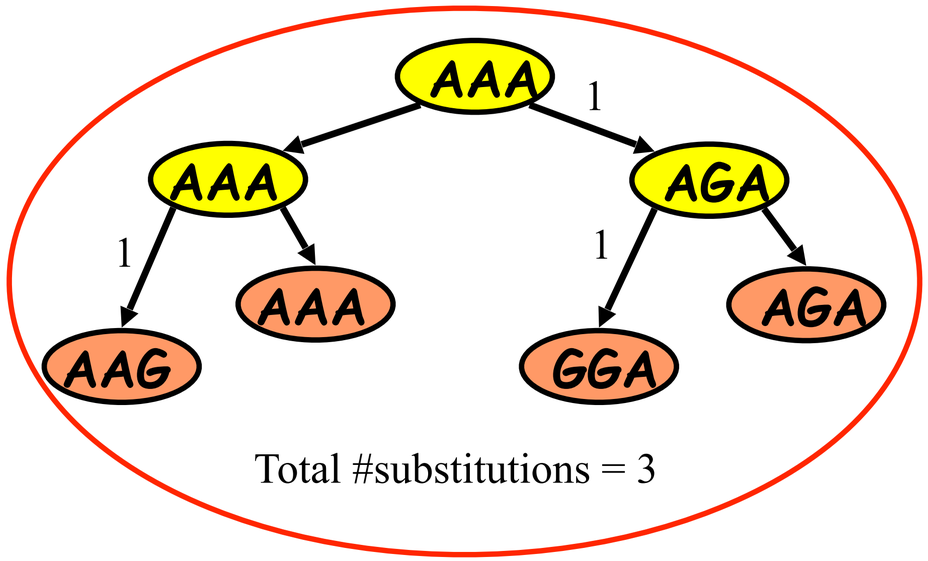
- In distance-based tree reconstruction algorithms, a certain amount of information gets lost in the transformation of the alignment matrix into the distance matrix, rendering the reverse transformation of distance matrix back into the alignment matrix impossible.
- A better technique is to use the alignment matrix directly for evolutionary tree reconstruction.
- Character-based tree reconstruction algorithms assume that the input data are described by an $n \times m$ matrix (perhaps an alignment matrix), where n is the number of species and m is the number of characters.

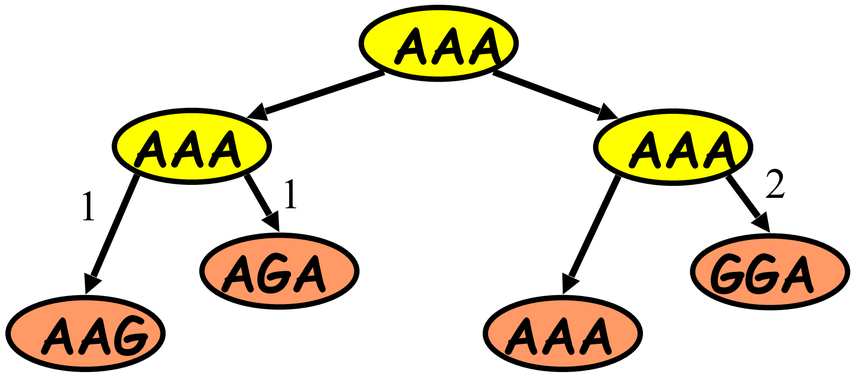
The principle of parsimony

- Occam's razor:" plurality should not be assumed without necessity," and is usually paraphrased as "keep it simple, stupid."
- Aristotle wrote simply that Nature operates in the shortest way possible.
- For character-based tree construction, the parsimony approach attempts to minimize the total number of mutations required to explain all of the observed character sequences.

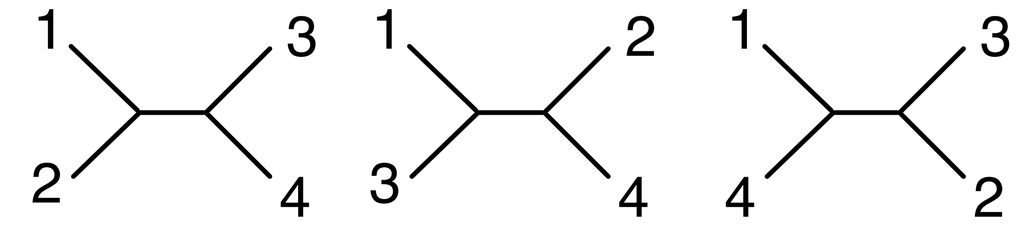
Maximum Parsimony

- Input: four nucleotide sequences: AAG, AAA, GGA, AGA taken from four species.
- Question: Which evolutionary tree best explains these sequences ?
- One Answer (the parsimony principle): Pick a tree that has a minimum total number of substitutions of symbols between species and their originator in the phylogenetic tree.

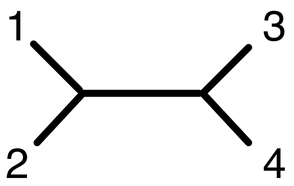




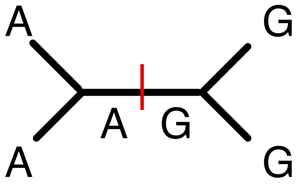
	1	2	3	4	5	6	7	8	9	10
Species 1	A	G	G	G	T	A	A	C	T	G
Species 2	A	C	G	A	T	T	A	T	T	A
Species 3	A	T	A	A	T	T	G	T	C	T
Species 4	A	A	T	G	T	T	G	T	C	G



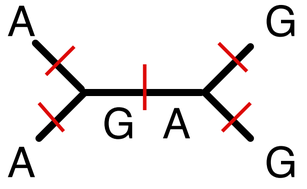
tree



1 change



5 changes

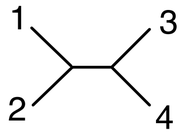


1 - A G G G T A A C T G

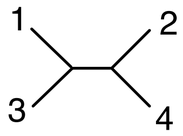
2 - A C G A T T A T T A

3 - A T A A T T G T C T

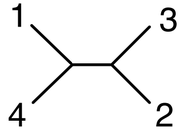
4 - A A T G T T G T C G



0



0



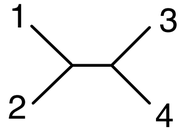
0

1 - A G G G T A A C T G

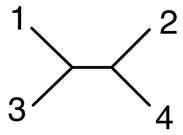
2 - A C G A T T A T T A

3 - A T A A T T G T C T

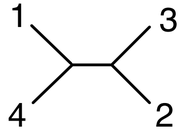
4 - A A T G T T G T C G



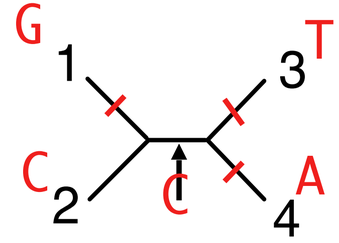
3



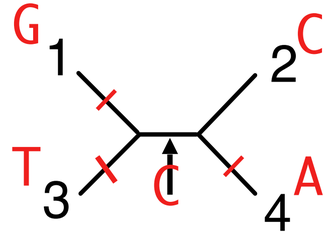
3



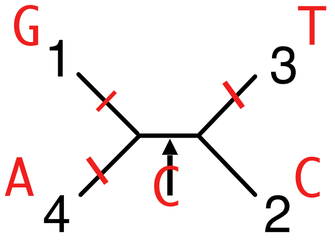
3



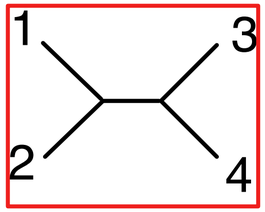
3



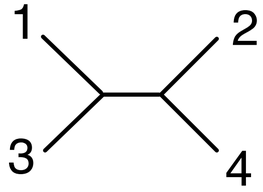
3



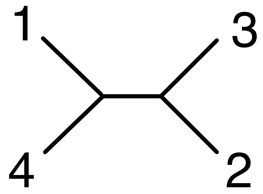
3



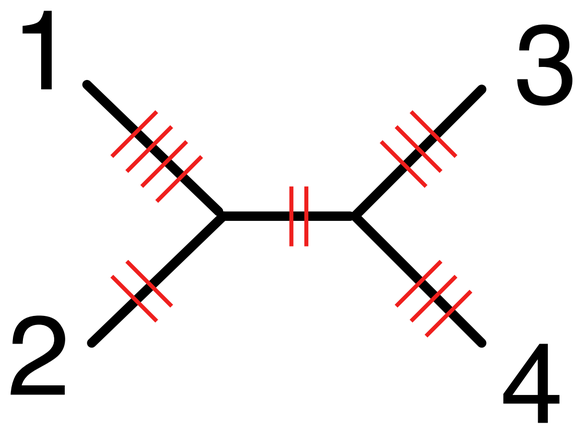
0 3 2 2 0 1 1 1 1 3 14



0 3 2 2 0 1 2 1 2 3 16



0 3 2 1 0 1 2 1 2 3 15



- Identify informative sites.
    - If a site is constant, then it is not informative. Informative sites are column positions in which there are at least two states (e.g., two different amino acid residues) with at least two taxa having each state.
- Construct trees.
    - Every tree is assigned a cost, and the tree with the lowest cost is sought. When necessary, a heuristic search is performed to reduce the complexity of the search by ignoring large families of trees that are unlikely to contain the shortest tree.
- Count the number of changes and select the shortest tree (or trees).

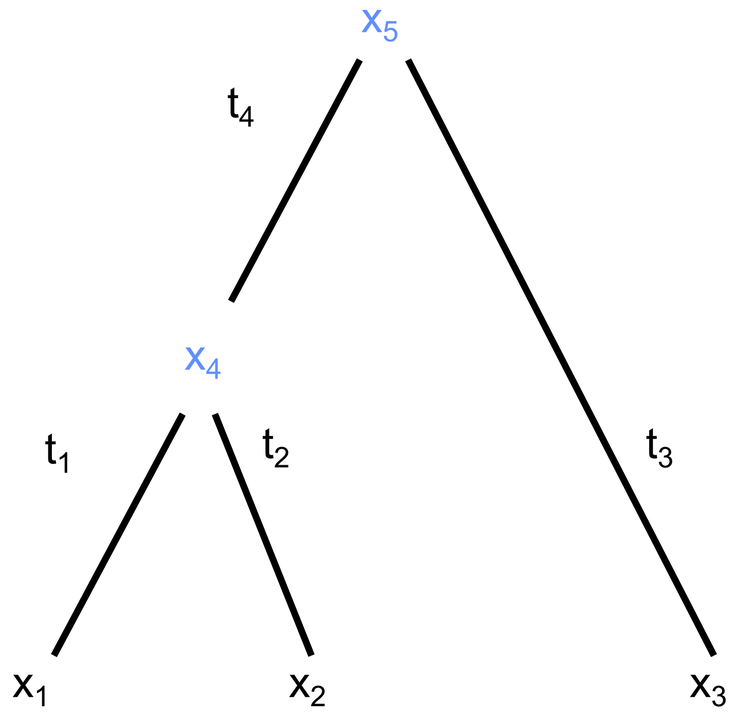
Heuristic approach for parsimony problem

- Heuristic methods are used to search tree space for most parsimonious trees
    - Techniques such as Nearest neighbor interchange (NNI), Subtree pruning and regrafting (SPR), Tree bisection and reconnection (TBR) are often applied to define neighbors in the space of all trees.
    - A greedy approach to the Large Parsimony problem is to start from an arbitrary tree and to move (by nearest neighbor interchange) from one tree to another if such a move provides the best improvement in the parsimony score among all neighbors of the tree T .

Maximum-likelihood Method

- The phylogenetic tree represents a generative probabilistic model (like HMMs) for the observed sequences.
  - Background probabilities: q(a)
  - Mutation probabilities: P(a|b, t)
  - Models for evolutionary mutations
    - Jukes Cantor
    - Kimura 2-parameter model
- Maximum likelihood is an approach that is designed to determine the tree topology and branch lengths that have the greatest likelihood of producing the observed dataset.

Probabilistic Approach



$$P(x_1, x_2, x_3, x_4, x_5 \mid T, t) =$$

$$q(x_5)p(x_4 \mid x_5, t_4)p(x_3 \mid x_5, t_3)p(x_1 \mid x_4, t_1)p(x_2 \mid x_4, t_2)$$

Computing the Tree Likelihood

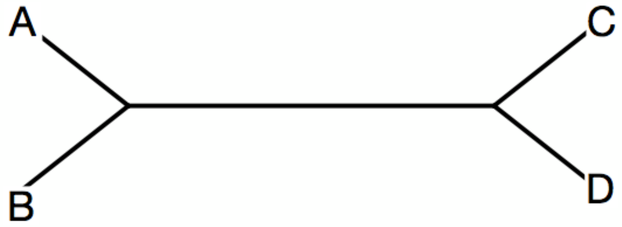
- We are interested in the probability of observed data given tree and branch “lengths”:
- Computed by summing over internal nodes
- This can be done efficiently using a tree upward traversal pass.

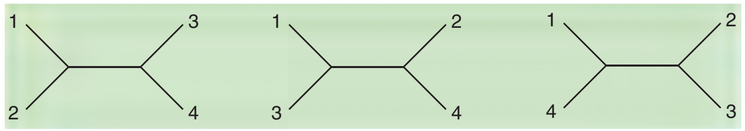
$$P(x1, x2, x3 \mid T, t) = \sum_{x4, x5} P(x1, x2, x3, x4, x5 \mid T, t)$$

Illustration for ML estimation of phylogenetic tree

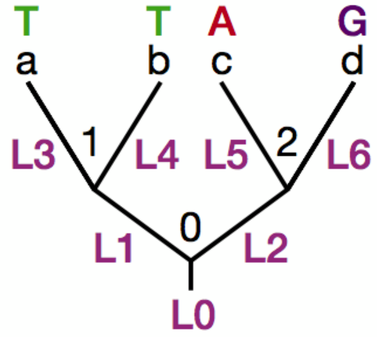
A. Sequences

B. An unrooted phylogenetic tree for the sequences A-D.





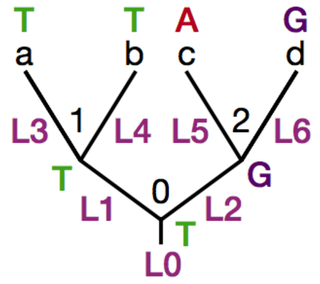
C. A rooted phylogenetic tree for the sequences A-D showing the bases for one set of aligned sequence positions in A.



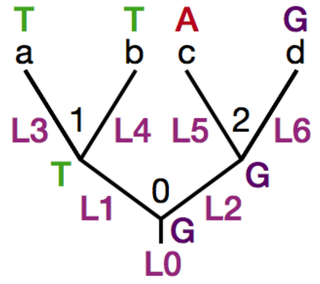
0, 1, 2: internal nodes

Illustration for ML estimation of phylogenetic tree

D. **A rooted phylogenetic tree showing one set of base assignments to nodes 0, 1 and 2.**



E. **A rooted phylogenetic tree showing a second set of base assignments to nodes 0, 1 and 2.**



F. **L(Tree) = L(Tree1) + L(Tree2) + ... + L(Tree64)**

- There are 4x4x4 combinations of internal node assignments!
- The likelihood of a tree given a particular choice of base assignments at node 0, 1, 2 is the product of the probability of node 0 times the product of each of the substitution probability: L0xL1xL2xL3xL4xL5xL6.
- In Fig. D, L0 = frequency of T (0.25), L2 (T->G) = probability of transversion ($2x10^{-6}$); L5 (G->A) = probability of transition (10^{-6}). All others have probability of 1. Then, L=0.25 x $2x10^{-6}$ x10^{-6} =$5x10^{-13.}$ ($2x10^{-6}$)
- The likelihood of the tree is the sum of the likelihood of all 64 combinations of trees.
- The final likelihood is the sum of the likelihood over all columns.

DISTANCE, PARSIMONY, AND MAXIMUM LIKELIHOOD: WHAT'S THE DIFFERENCE?

Distance matrix methods simply count the number of differences between two sequences. This number is referred to as the evolutionary distance, and its exact size depends on the evolutionary model used. The actual tree is then computed from the matrix of distance values by running a clustering algorithm that starts with the most similar sequences (i.e., those that have the shortest distance between them) or by trying to minimize the total branch length of the tree. The principle of maximum parsimony searches for a tree that requires the smallest number of changes to explain the differences observed among the taxa under study.

A maximum-likelihood approach to phylogenetic inference evaluates the probability that the chosen evolutionary model has generated the observed data. The evolutionary model could simply mean that one assumes that changes between all nucleotides (or amino acids) are equally probable. The program will then assign all possible nucleotides to the internal nodes of the tree in turn and calculate the probability that each such sequence would have generated the data (if two sister taxa have the nucleotide "A," a reconstruction that assumes derivation from a "C" would be assigned a low probability compared with a derivation that assumes there already was an "A"). The probabilities for all possible reconstructions (not just the more probable one) are summed up to yield the likelihood for one particular site. The likelihood for the tree is the product of the likelihoods for all alignment positions in the data set.

Evaluating trees: the Bootstrap

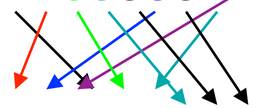
ATAGCCATA

ATACCCATG

ATACCCATA

ATAGCCATA

ATCCCCCAT



TCAAATGCA

TCGAATCCA

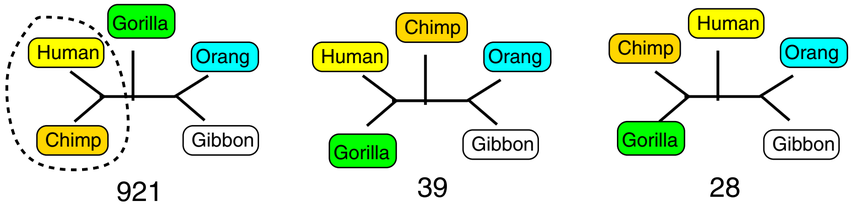
TCAAATCCA

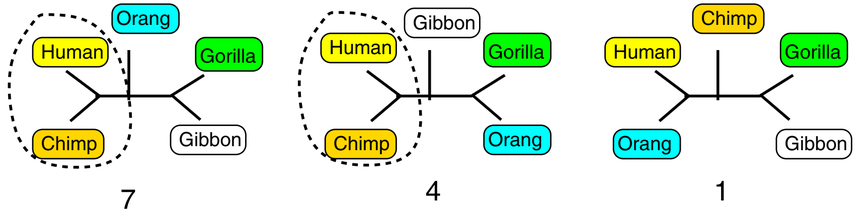
TCAAATGCA

TCAACACCC

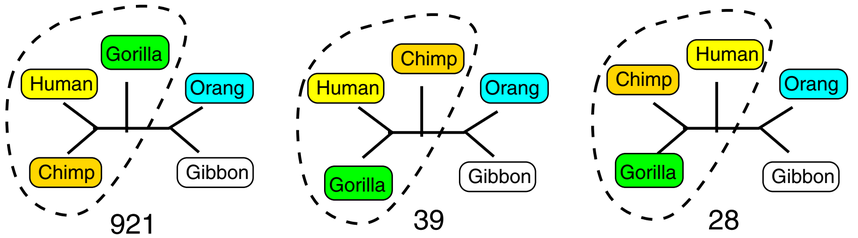
- Often we don't trust the tree found as the "correct" one.
- Bootstrapping:
  - Sample (with replacement) n positions from the alignment
  - Learn the best tree for each sample
  - Look for tree features which are frequent in all trees.
- In bootstrap analysis, multiple datasets are created by randomly sampling with replacement from the original dataset. This simulates the variability in the data and allows for the assessment of how robust the inferred relationships are.
- Each bootstrap replicate generates a phylogenetic tree. By comparing the trees, researchers can determine the frequency with which specific clades (groups of organisms) appear across the bootstrap replicates. A higher frequency indicates stronger support for that clade.
- Bootstrap values provide insights into the stability of the tree structure. High bootstrap support for a branch suggests that it is likely a reliable inference, while low support may indicate that the branch is sensitive to changes in the data.

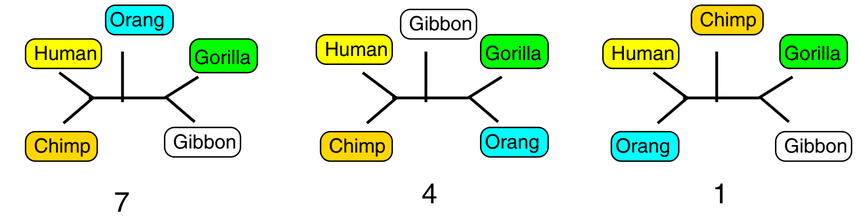
Bootstrap support value



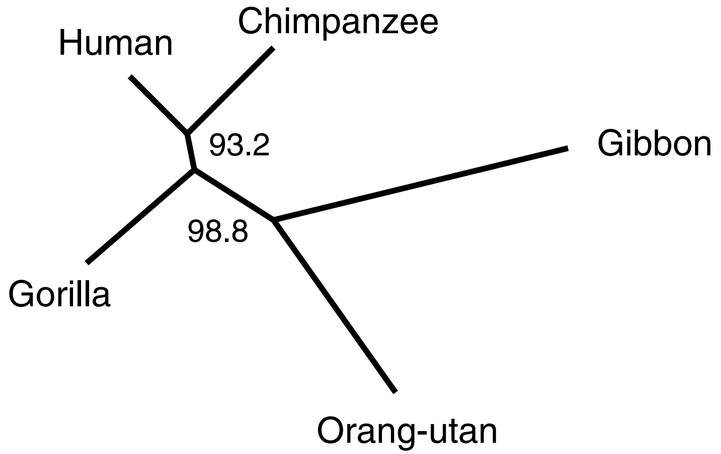


P= (921+7+4)/1000= 0.932





P= (921+39+28)/1000= 0.988



PHYLIP Home Page

evolution.genetics.washington.edu/phylip.html

Reader

苹果中国 iCloud 新浪微博 腾讯微博 维基百科 百度 中国雅虎 欢迎访问复旦...财务信息门户 Handbook of... dispersion Apple Facebook Twitter Wikipedia Yahoo! News

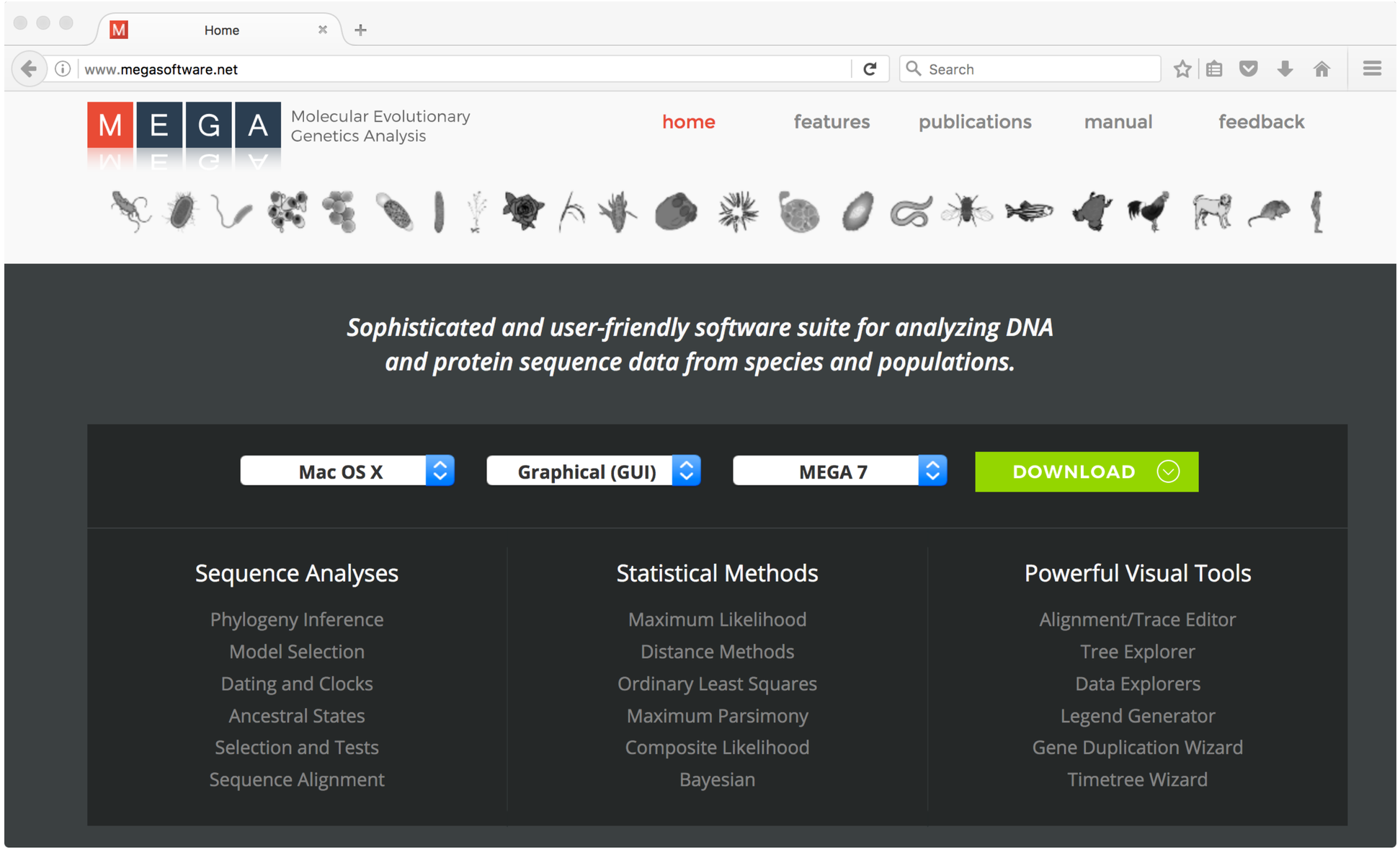
PHYLIP

PHYLIP has a Facebook page here. It is intended to be used for discussion of bugs, fixes, new features, and how to use the package.

PHYLIP is a *free* package of programs for inferring phylogenies. It is distributed as source code, documentation files, and a number of different types of executables. These Web pages, by Joe Felsenstein of the Department of Genome Sciences and the Department of Biology at the University of Washington, contain information on PHYLIP and ways to transfer the executables, source code and documentation to your computer.

- A general description of PHYLIP.
- Programs in the PHYLIP package
- About the Executables
- About the Source code ... compiling it yourself
- The documentation web pages for PHYLIP can be read here
- Get me PHYLIP (version 3.695)
- How to install PHYLIP
- Frequently asked questions
- PHYLIP's Facebook page for discussing problems.
- An excellent guide to using PHYLIP with molecular data is available here.
- PHYLIP on the web (HTML documentation, server services)
- Current and future versions of PHYLIP (including new features)
- Older versions of PHYLIP, including version 3.5
- Bugs in the package, known or recently fixed
- Phylogeny programs available elsewhere
- Credits (people, grants etc.)

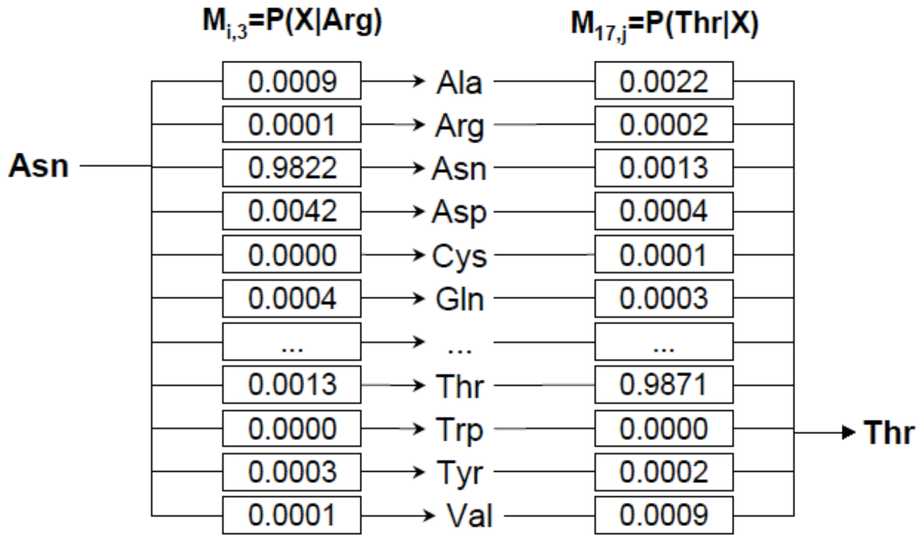
This is access number 16708 of this page since 1 January 2014. To see the number of accesses in previous years look here.


Home
www.megasoftware.net
Search
MEGA
Molecular Evolutionary Genetics Analysis
home
features
publications
manual
feedback
Sophisticated and user-friendly software suite for analyzing DNA and protein sequence data from species and populations.
Mac OS X
Graphical (GUI)
MEGA 7
DOWNLOAD
Sequence Analyses
Phylogeny Inference
Model Selection
Dating and Clocks
Ancestral States
Selection and Tests
Sequence Alignment
Statistical Methods
Maximum Likelihood
Distance Methods
Ordinary Least Squares
Maximum Parsimony
Composite Likelihood
Bayesian
Powerful Visual Tools
Alignment/Trace Editor
Tree Explorer
Data Explorers
Legend Generator
Gene Duplication Wizard
Timetree Wizard

The use of Markov chain when constructing the mutation probability matrix for the evolutionary distance of PAM2

A **Markov process** is one where the likelihood of the next "state" depends only on the current state. The inference that **evolution** is **Markovian** assumes that base changes (or amino acid changes) occur at a constant rate and depend only on the *identity of the current base (or amino acid)*.

From PAM1 to PAM2



P(Asn -> Thr)= P(Asn -> Ala -> Thr) + P(Asn -> Arg -> Thr) + ... + P(Asn -> Val -> Thr)
= (0.0009)(0.0001) + (0.0001)(0.0002) + ... + (0.0001)(0.009)

line 3 of PAM1 column 17 of PAM1

=> Matrix product: PAM2 = PAM1 x PAM1

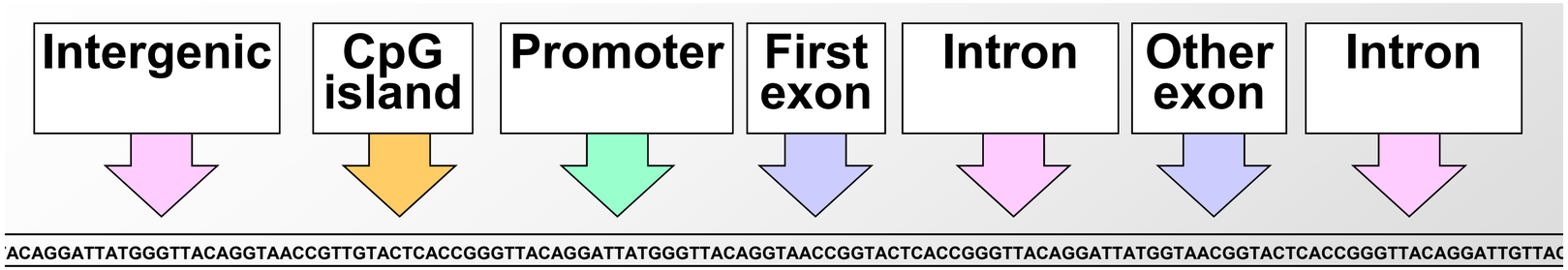
Biological problems and HMM

Often, problems in biological sequence analysis are just a matter of putting the right label on each residue.

In gene identification, we want to label nucleotides as exons, introns, or intergenic sequence.

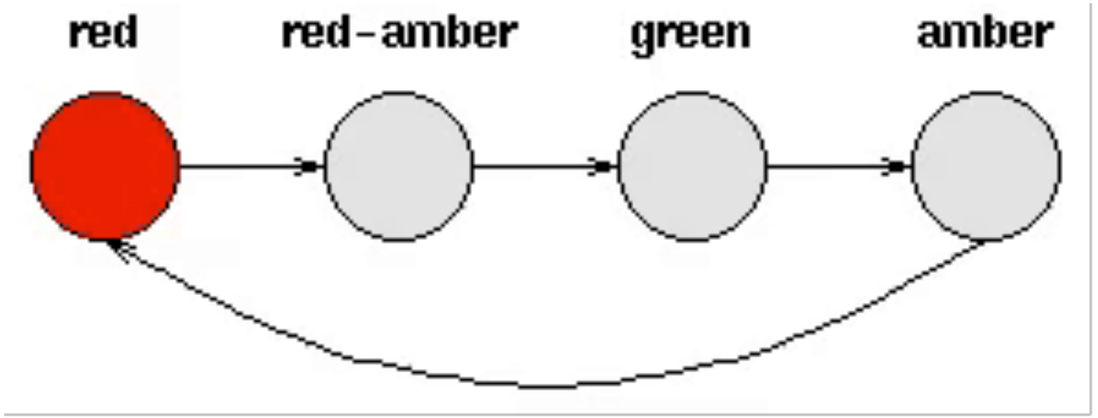
In sequence alignment, we want to associate residues in a query sequence with homologous residues in a target database sequence.

Hidden Markov Model provides a probabilistic model for putting labels to biological sequences.



A simple Markov chain

Traffic light



The next traffic light is solely dependent on the previous traffic light:

Red -> red-amber

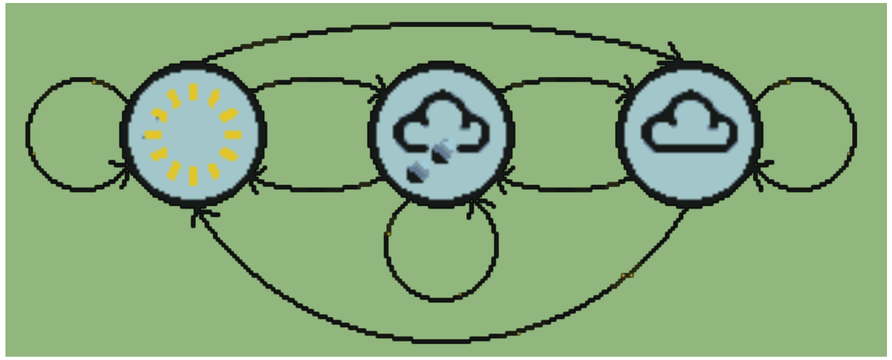
Red-amber -> green

Green -> amber

A Markov chain is a mathematical system that undergoes transitions from one state to another on a state space. It is a random process usually characterized as **memoryless**: ***the next state depends only on the current state and not on the sequence of events that preceded it***. This specific kind of "**memorylessness**" is called the ***Markov property***.

Markov chain

Weather change



Yesterday \ Today	*sun*	*cloud*	*rain*
sun	0.50	0.375	0.125
cloud	0.25	0.125	0.625
rain	0.25	0.375	0.375

=1

***Markov assumption*:** weather can be predicted from current weather state.

There are three weather states: sun, cloud, rain.

The transition from one state to another state follows a transition probability.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2N} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{iN} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{Nj} & \dots & a_{NN} \end{bmatrix}$$

$$a_{ij} = P(q_t = \mathrm{j} | \mathrm{q}_{t-1} = i) \quad 1 \le i, j \le N$$

$$a_{ij} \ge 0, \quad \forall i, j$$

$$\sum_{j=1}^{N} a_{ij} = 1, \quad \forall i$$

The probability of a Markov chain

For any probabilistic model of sequence **X**, the probability of the sequence is:

$$P(A, B) = P(A|B)P(B)$$

$$\begin{aligned}
P(x) &= P(x_1, x_2, ..., x_n) \\
&= P(x_n|x_1, x_2, ..., x_{n-1})P(x_1, x_2, ..., x_{n-1}) \\
&= P(x_n|x_1, x_2, ..., x_{n-1})P(x_{n-1}|x_1, x_2, ..., x_{n-2})P(x_1, x_2, ..., x_{n-2}) \\
&... \\
&= P(x_n|x_1, x_2, ..., x_{n-1})P(x_{n-1}|x_1, x_2, ..., x_{n-2})...P(x_1) \\
&= P(x_n|x_{n-1})P(x_{n-1}|x_{n-2})..P(x_2|x_1)P(x_1) \qquad \textbf{Markov chain} \\
&= P(x_1)\prod_{i=2}^{n} a_{x_{i-1}x_i} \\
&= \prod_{i=1}^{n} a_{x_{i-1}x_i}
\end{aligned}$$

By adding a begin state x_0, $P(x_1) = P(x_1 \mid x_0)$

Markov chain formalization

- A Markov Model M = (Q, A, p)
- **Q**: Represents the set of states in the system. Each state corresponds to a specific condition or configuration. ((sun, cloudy, rain)
- **A**: Represents the transition probabilities, typically organized in a transition matrix. Each entry in this matrix indicates the probability of transitioning from one state to another.
- **p**: Represents the initial state distribution, indicating the probabilities of starting in each state when the process begins.

Markov property:

$$Pr(X_{n+1} = j | X_0 = q_0, ..., X_{n-1} = q_{n-1}, X_n = i) = Pr(X_{n+1} = j | X_n = i) = a_{i,j}$$

transition probability

The ***future*** behavior of the system depends only on the ***current*** state i and not on any of the previous states.

- A Markov Model M = (Q, A, p)

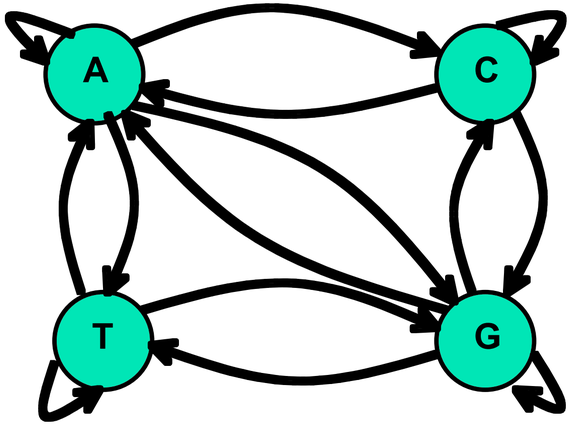
- **Q**: Represents the set of states in the system. Each state corresponds to a specific condition or configuration. ((sun, cloudy, rain)

- **A**: Represents the transition probabilities, typically organized in a transition matrix. Each entry in this matrix indicates the probability of transitioning from one state to another.

- **p**: Represents the initial state distribution, indicating the probabilities of starting in each state when the process begins.

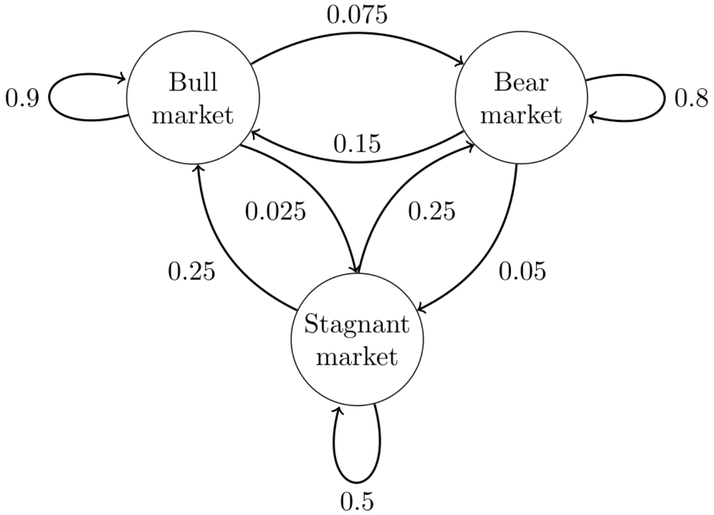
Markov chain examples

A Markov chain of DNA sequence from CpG islands



+	A	C	G	T
A	0.180	0.274	0.426	0.120
C	0.171	0.368	0.274	0.188
G	0.161	0.339	0.375	0.125
T	0.079	0.355	0.384	0.182

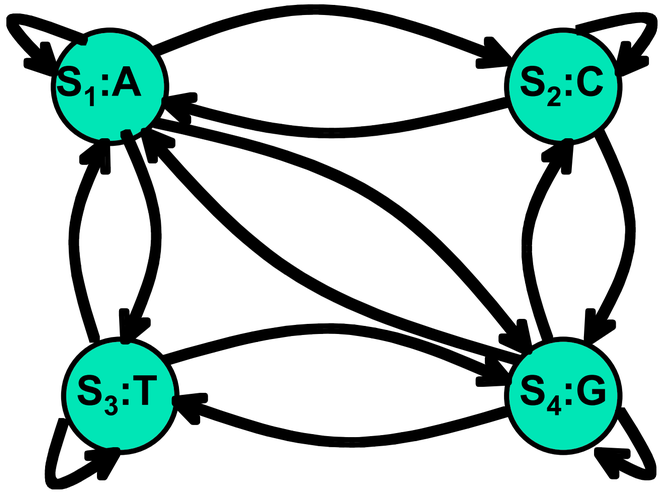
A Markov chain of stock market



$$P = \begin{array}{c} \begin{matrix} \text{Bull} & \text{Bear} & \text{Stagnent} \end{matrix} \\ \begin{bmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{bmatrix} \end{array} \begin{matrix} \text{Bull} \\ \text{Bear} \\ \text{Stagnent} \end{matrix}$$

The probability of a Markov chain using CpG island model (+)

Sequence: GAGCGCGTAC



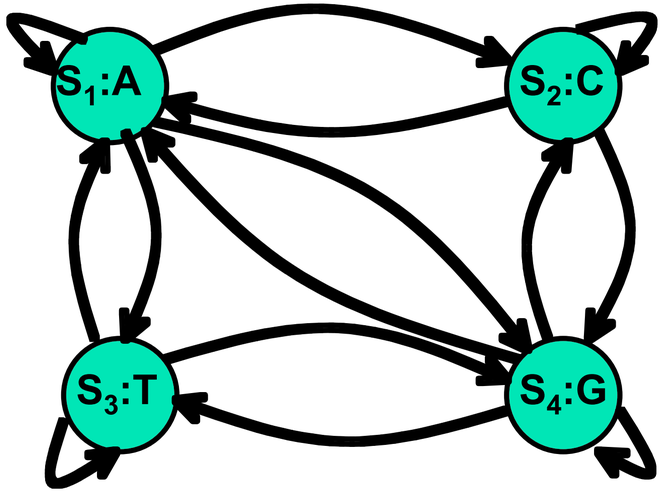
States: $S_4S_1S_4S_2S_4S_2S_4S_3S_1S_2$

+	A	C	G	T
A	0.180	0.274	0.426	0.120
C	0.171	0.368	0.274	0.188
G	0.161	0.339	0.375	0.125
T	0.079	0.355	0.384	0.182

$$
\begin{aligned}
P &= \prod_{i=1}^{n} a_{x_{i-1}x_i} \\
&= P(G)P(A|G)P(G|A)P(C|G)..P(C|A) \\
&= 1 \times 0.161 \times 0.426 \times 0.339 \times 0.274 \times 0.339 \times 0.274 \times 0.125 \times 0.079 \times 0.274 \\
&= 1.601119195778 92e-06
\end{aligned}
$$

The probability of a Markov chain using non-CpG island model (-)

Sequence: GAGCGCGTAC



States: $S_4S_1S_4S_2S_4S_2S_4S_3S_1S_2$

–	A	C	G	T
A	0.300	0.205	0.285	0.210
C	0.322	0.298	0.078	0.302
G	0.248	0.246	0.298	0.208
T	0.177	0.239	0.292	0.292

$$P = \prod_{i=1}^{n} a_{x_{i-1}x_i}$$

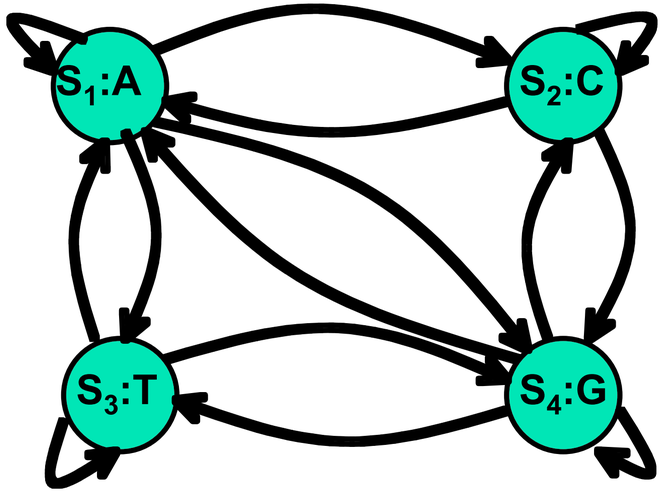
$$= P(G)P(A|G)P(G|A)P(C|G)..P(C|A)$$

$$= 1 \times 0.248 \times 0.285 \times 0.246 \times 0.078 \times 0.246 \times 0.078 \times 0.208 \times 0.177 \times 0.205$$

$$= 1.964022337244 84e - 07$$

Which model fits better the sequence?

Sequence: GAGCGCGTAC



States: $S_4S_1S_4S_2S_4S_2S_4S_3S_1S_2$

$$S(x) = \log \frac{P(x|\text{model}+)}{P(x|\text{model}-)} = \sum_{i=1}^{L} \log \frac{a^{+}_{x_{i-1}x_i}}{a^{-}_{x_{i-1}x_i}}$$

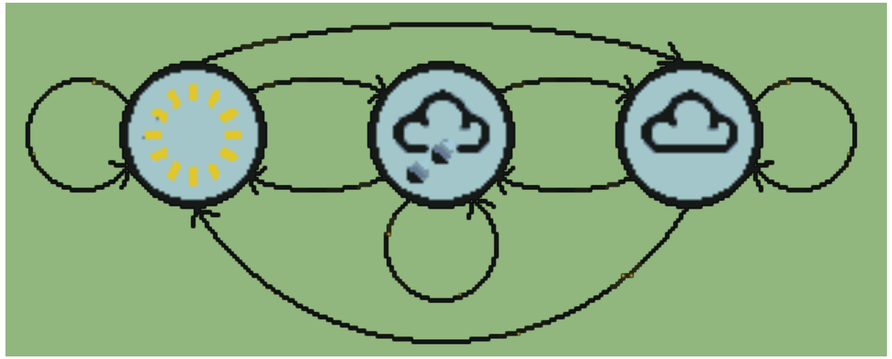
$$= 2.098$$

The sequence more likely comes from the CpG island model.

What is a “hidden” Markov Model?

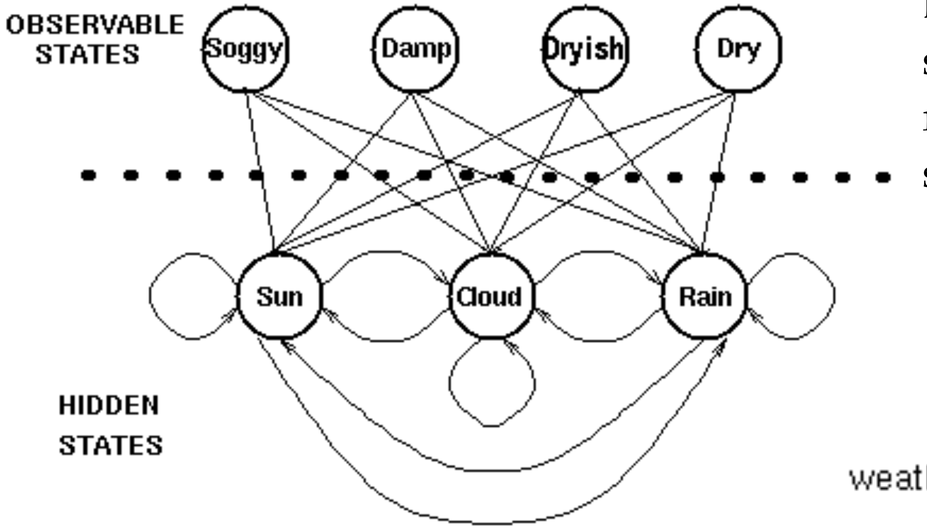
Hidden Markov Model

Weather change



transition probability

Yesterday \ Today	*sun*	*cloud*	*rain*
sun	0.50	0.375	0.125
cloud	0.25	0.125	0.625
rain	0.25	0.375	0.375



Suppose weather cannot be directly observed. But the degree of wetness (dry, dryish, damp, soggy) of seeweed has been thought to be related to weather condition. In this case, the state of weather is "hidden".

emission probability

seaweed

weather	Dry	Dryish	Damp	Soggy
Sun	0.60	0.20	0.15	0.05
Cloud	0.25	0.25	0.25	0.25
Rain	0.05	0.10	0.35	0.50

Formal definition of Hidden Markov Model

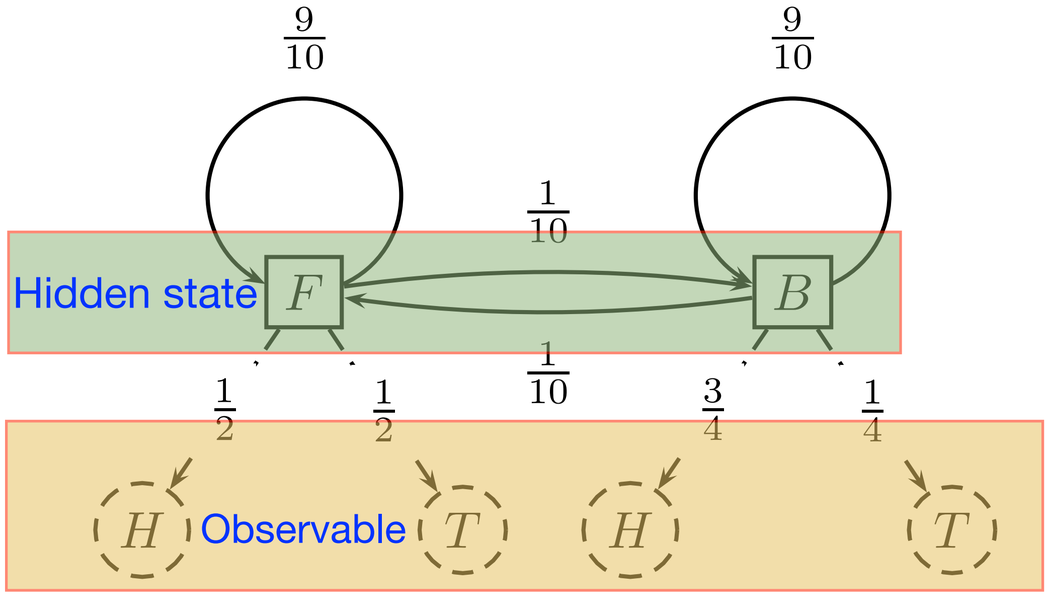
Formally, an HMM $\mathcal{M}$ is defined by an alphabet of emitted symbols Σ, a set of (hidden) states Q, a matrix of state transition probabilities A, and a matrix of emission probabilities E, where

- Σ is an alphabet of symbols;
- Q is a set of states, each of which will emit symbols from the alphabet Σ;
- $A = (a_{kl})$ is a $|Q| \times |Q|$ matrix describing the probability of changing to state l after the HMM is in state k; and
- $E = (e_k(b))$ is a $|Q| \times |\Sigma|$ matrix describing the probability of emitting the symbol b during a step in which the HMM is in state k.

The occasionally dishonest casino and Hidden Markov Model

The “Fair Bet Casino” has a game in which a dealer flips a coin and the player bets on the outcome (heads or tails)

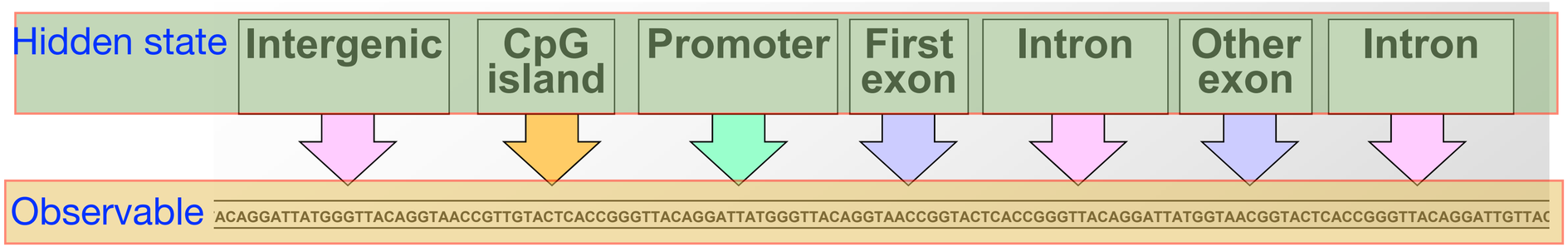
The dealer in this casino uses either a fair coin or a biased coin that will give heads with a probability of $3/4$. The dealer change coins with certain probability.



Given the observed sequence, can we infer the hidden state? i.e., when did the dealer use the biased coin?

0 1 0 1 1 1 0 1 0 0 1

Genomic structure and Hidden Markov Model



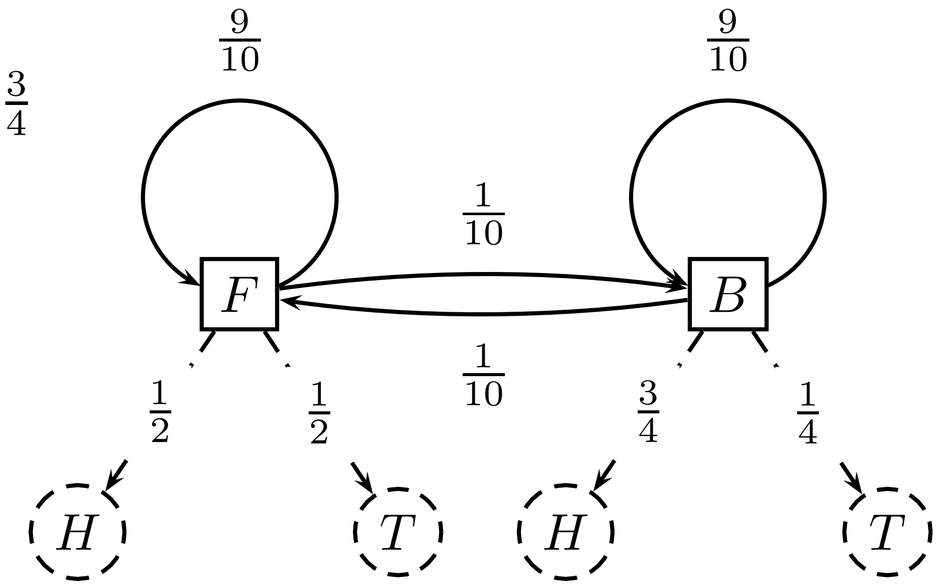
Given the observed DNA sequence, can we infer the hidden state (exon, intron, …)?

- Given a HMM *M* and an observed sequence *x*,
    1. What is the most probable path of states for *x*?
    2. What is the probability that the i^{th} sequence of x being state k?
- Given a set of sequences, how can we estimate *M*?

The occasionally dishonest casino HMM

Bet Casino process corresponds to the following HMM $\mathcal{M}(\Sigma, Q, A, E)$ shown in figure 11.1:

- $\Sigma = \{0, 1\}$, corresponding to tails (0) or heads (1)
- $Q = \{F, B\}$, corresponding to a fair (F) or biased (B) coin
- $a_{FF} = a_{BB} = 0.9, a_{FB} = a_{BF} = 0.1$
- $e_F(0) = \frac{1}{2}, e_F(1) = \frac{1}{2}, e_B(0) = \frac{1}{4}, e_B(1) = \frac{3}{4}$



HMM path

- A Hidden Markov Model (HMM) path refers to a specific sequence of states that the model transitions through over time, where these states are not directly observable (hidden).

$$\begin{matrix} x \\ \pi \\ P(x_i|\pi_i) \end{matrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ F & F & F & B & B & B & B & B & F & F & F \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

$P(x_i \mid \pi_i)$: the probability that symbol x_i was emitted from state π_i.

$P(\pi_i \rightarrow \pi_{i+1})$: the probability of the transition from state π_i to π_{i+1}.

- A Hidden Markov Model (HMM) path refers to a specific sequence of states that the model transitions through over time, where these states are not directly observable (hidden).

Probability of an observed sequence given an HMM path

$$
\begin{array}{c}
x \\ \pi \\ P(x_i|\pi_i) \\ P(\pi_{i-1} \to \pi_i)
\end{array}
=
\begin{pmatrix}
0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
F & F & F & B & B & B & B & B & F & F & F \\
\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10}
\end{pmatrix}
$$

$$
\begin{aligned}
P(x,\pi) &= \left(\tfrac{1}{2}\cdot\tfrac{1}{2}\right)\left(\tfrac{1}{2}\cdot\tfrac{9}{10}\right)\left(\tfrac{1}{2}\cdot\tfrac{9}{10}\right)\left(\tfrac{3}{4}\cdot\tfrac{1}{10}\right)\left(\tfrac{3}{4}\cdot\tfrac{9}{10}\right)\left(\tfrac{3}{4}\cdot\tfrac{9}{10}\right)\left(\tfrac{1}{4}\cdot\tfrac{9}{10}\right)\left(\tfrac{3}{4}\cdot\tfrac{9}{10}\right)\left(\tfrac{1}{2}\cdot\tfrac{1}{10}\right)\left(\tfrac{1}{2}\cdot\tfrac{9}{10}\right)\left(\tfrac{1}{2}\cdot\tfrac{9}{10}\right) \\
&= 2.66 \times 10^{-6}
\end{aligned}
$$

The probability that sequence x was generated by the path π, given the model $\mathcal{M}$, is

$$
P(x|\pi) = P(\pi_0 \to \pi_1) \cdot \prod_{i=1}^{n} P(x_i|\pi_i) P(\pi_i \to \pi_{i+1}) = a_{\pi_0,\pi_1} \cdot \prod_{i=1}^{n} e_{\pi_i}(x_i) \cdot a_{\pi_i,\pi_{i+1}}.
$$

π_0 and π_{n+1} as the fictitious initial and terminal states *begin* and *end*.

What is the most probable path?

Decoding Problem:
Find an optimal hidden path of states given observations.

Input: Sequence of observations $x = x_1 \dots x_n$ generated by an HMM $\mathcal{M}(\sum, Q, A, E)$.

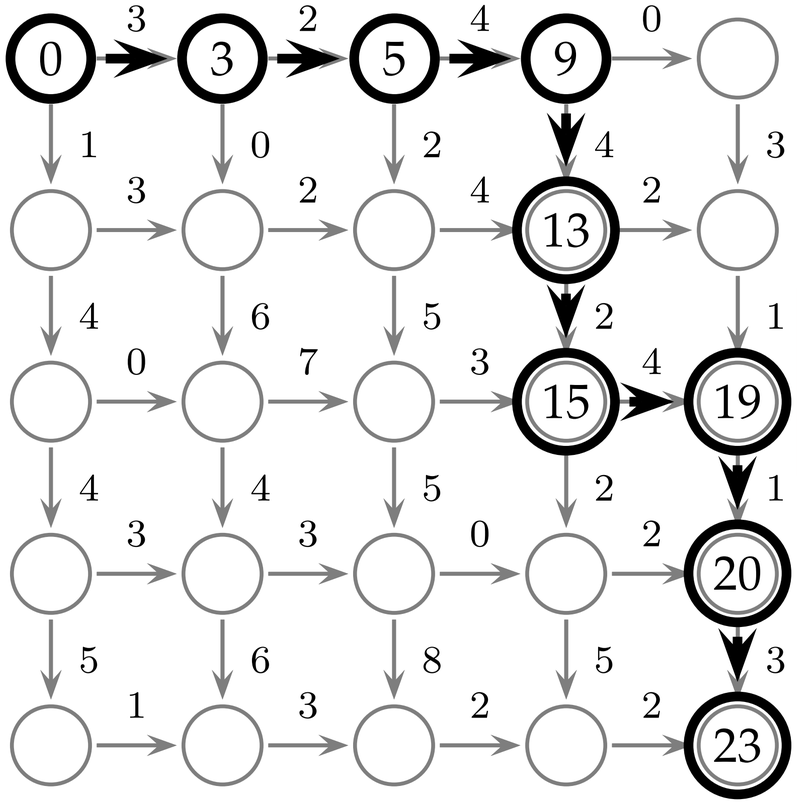
Output: A path that maximizes $P(x|\pi)$ over all possible paths π.

- One approach is to enumerate all possible paths and calculate the probability for each one, identifying the path with the highest probability. However, this method becomes impractical due to the exponential growth in the number of paths.

- So, how can we effectively address this problem?

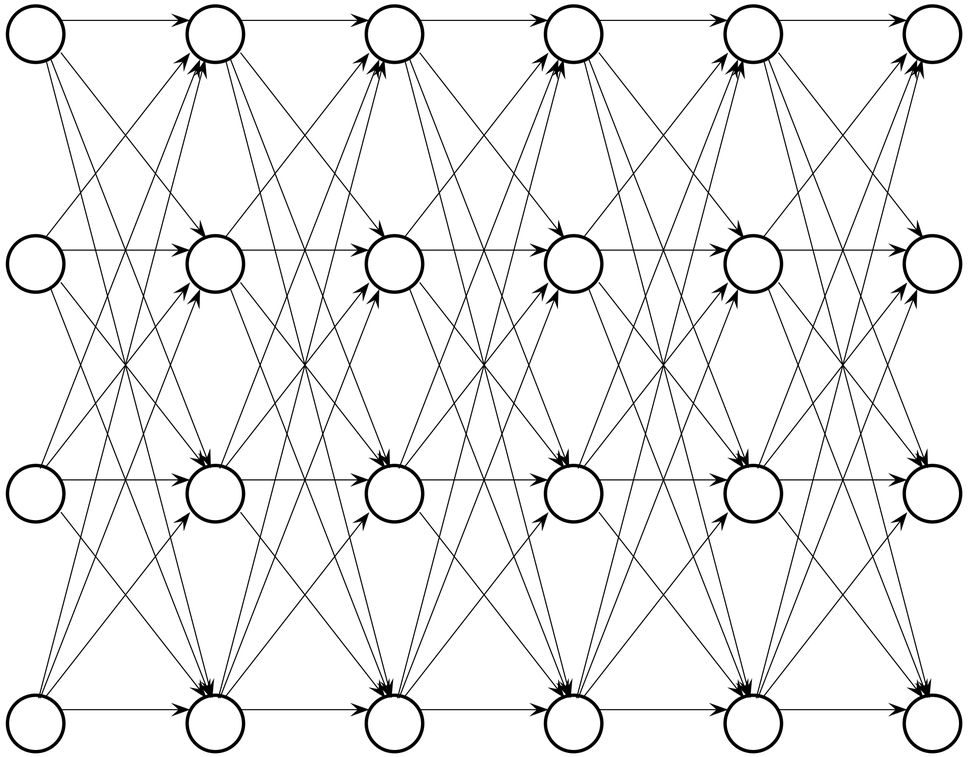
Revisit the Manhattan Tourist Problem

Manhattan Tourist Problem



$|Q|$ states

HMM path problem



observed symbols

n columns

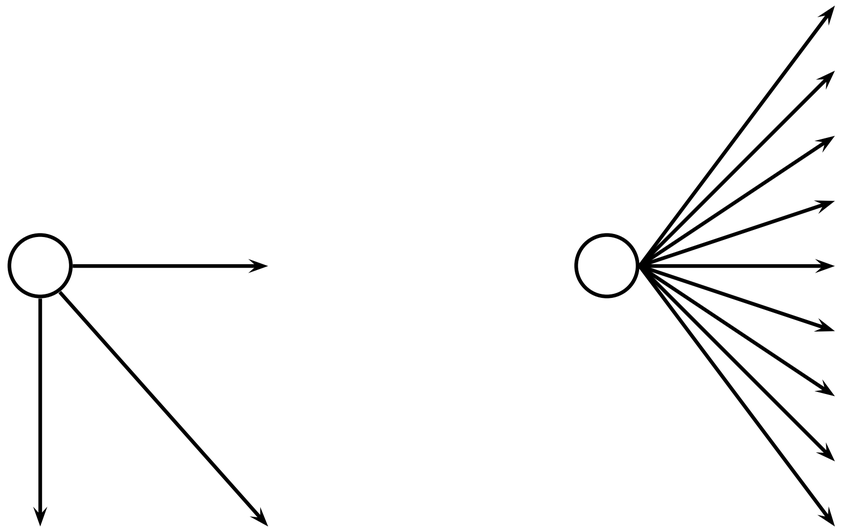
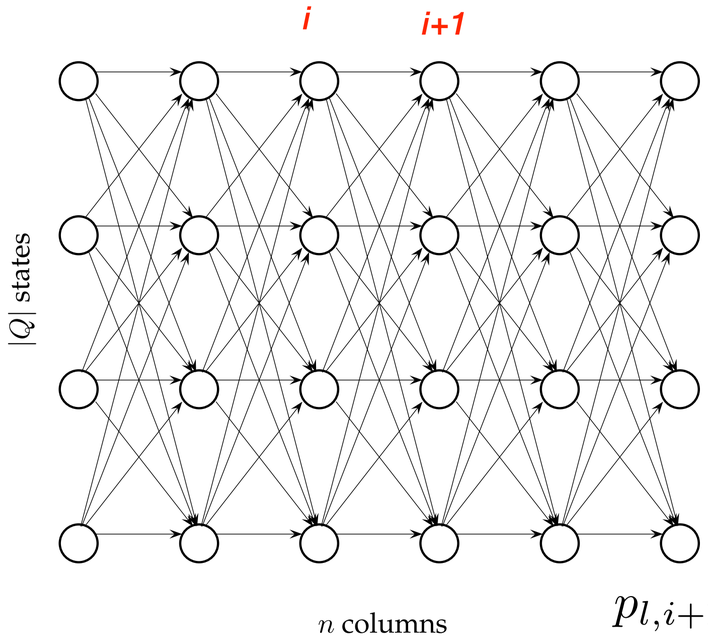


Figure 11.3 The set of valid directions in the alignment problem (a) is usually limited to south, east, and southeast edges, while the set of valid directions in the decoding problem (b) includes any eastbound edge.

The Viterbi Algorithm



- $P(x|\pi)$ = the product of the edge weights for path $\pi=\pi_1 \ldots \pi_n$.
- The weight of an edge from (k, i) to $(l, i+1) = e_l(x_{i+1}) \cdot a_{kl}$.
- $p_{k,i}$: the probability of a path ending in vertex (k, i).

$$
\begin{aligned}
p_{l,i+1} &= \prod_{j=1}^{i+1} e_{\pi_j}(x_j) \cdot a_{\pi_{j-1},\pi_j} \\
&= \left(\prod_{j=1}^{i} e_{\pi_j}(x_j) \cdot a_{\pi_{j-1},\pi_j} \right) \cdot \left(e_{\pi_{i+1}}(x_{i+1}) \cdot a_{\pi_i,\pi_{i+1}} \right) \\
&= p_{k,i} \cdot e_l(x_{i+1}) \cdot a_{kl} \\
&= p_{k,i} \cdot \text{weight of edge from } (k,i) \text{ to } (l,i+1)
\end{aligned}
$$

The Viterbi Algorithm

$$
\begin{aligned}
s_{l,i+1} &= \max_{k \in Q}\{s_{k,i} \cdot \text{ weight of edge between } (k,i) \text{ and } (l,i+1)\ \} \\
&= \max_{k \in Q}\{s_{k,i} \cdot a_{kl} \cdot e_l(x_{i+1})\} \\
&= e_l(x_{i+1}) \cdot \max_{k \in Q}\{s_{k,i} \cdot a_{kl}\}
\end{aligned}
$$

$s_{begin,0} = 1$ and $s_{k,0} = 0$ for $k \neq begin$.

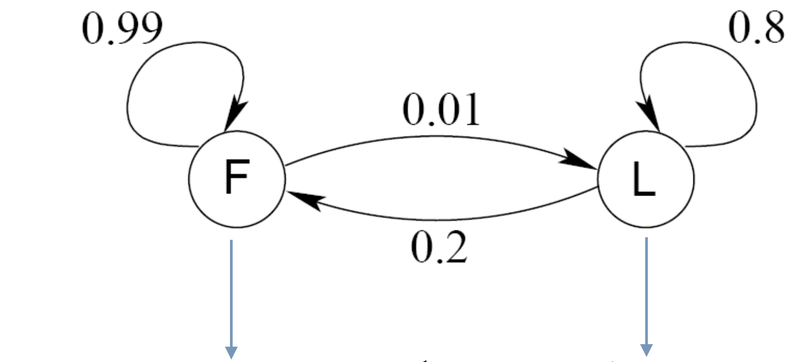
$$P(x|\pi^*) = \max_{k \in Q}\{s_{k,n} \cdot a_{k,end}\}$$

π^* is an optimal path

$$S_{l,i+1} = \log e_l(x_{i+1}) + \max_{k \in Q}\{S_{k,i} + \log(a_{kl})\}.$$

- The Viterbi algorithm efficiently finds the most likely sequence of hidden states in an HMM by breaking down the problem into manageable subproblems, characteristic of dynamic programming techniques.
  - **Optimal Substructure**: It leverages the principle of optimal substructure, meaning that the optimal path to a given state can be constructed from optimal paths to previous states.
  - **Recursive Calculation**: The algorithm recursively calculates the highest probability of reaching each state at each time step, storing these values to avoid redundant calculations.
  - **Backtracking**: Once the highest probabilities are computed, the algorithm backtracks through the stored values to reconstruct the most probable sequence of hidden states.

The Viterbi example



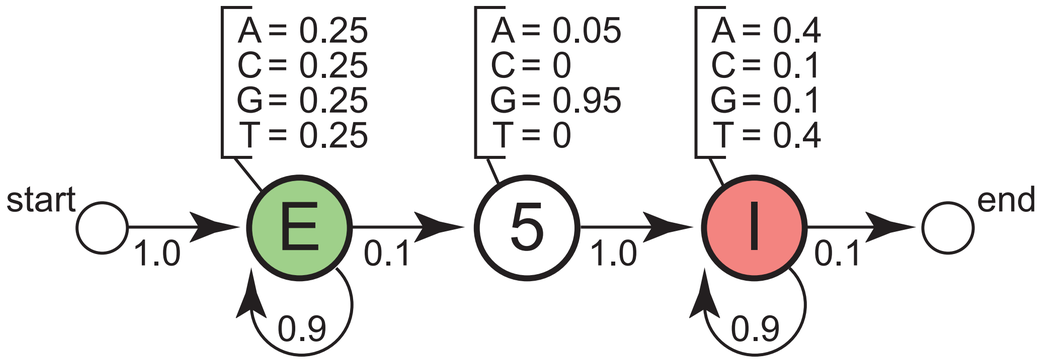
$$s_{l,i+1} = e_l(x_{i+1}) \cdot \max_{k \in Q} \{s_{k,i} \cdot a_{kl}\}$$

		ε	6	2	6
			x		
	B	1	0	0	0
π	F	0	(1/6)×(1/2) = 1/12	(1/6)×max{(1/12)×0.99, (1/4)×0.2} = 0.01375	(1/6)×max{0.01375×0.99, 0.02×0.2} = 0.00226875
	L	0	(1/2)×(1/2) = 1/4	(1/10)×max{(1/12)×0.01, (1/4)×0.8} = 0.02	(1/2)×max{0.01375×0.01, 0.02×0.8} = 0.08

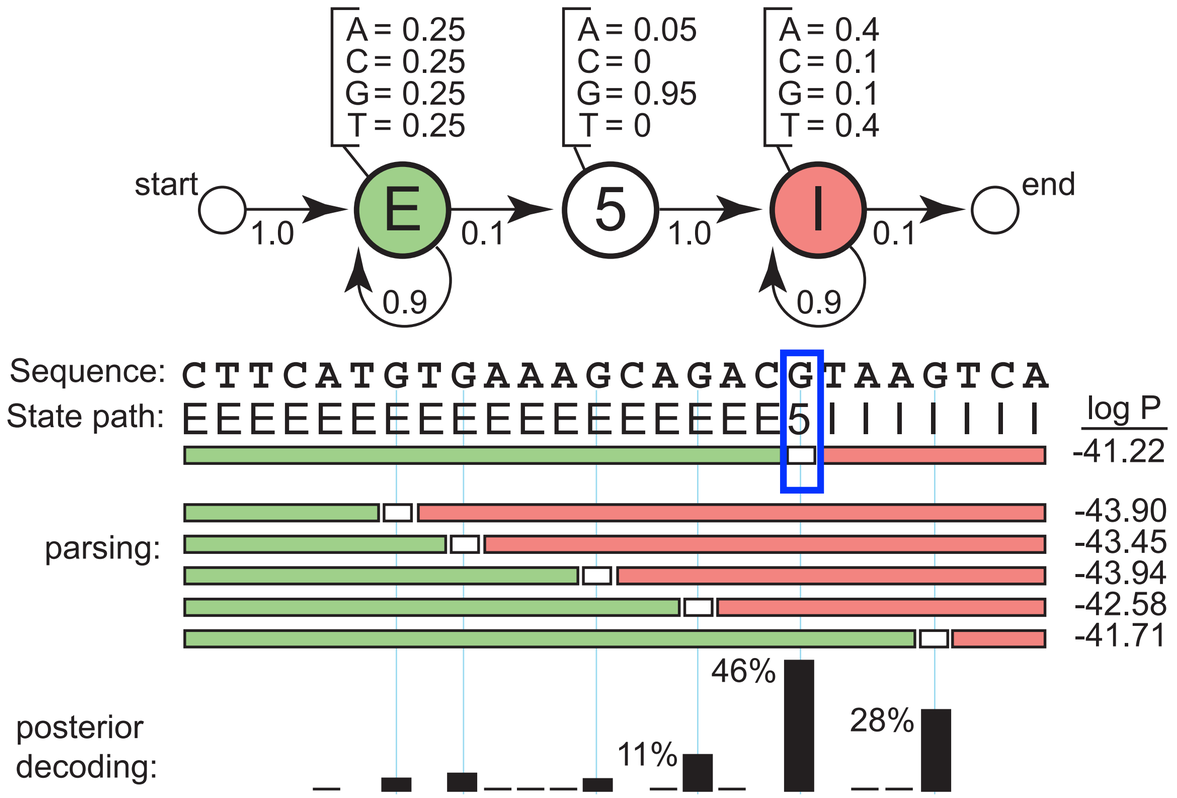
Viterbi gets it right more often than not

```
Rolls      3151162464466442453211316311641521336251445436316566266566666
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLL

Rolls      651166453132651245636664631636663162326455235266666625151631
Die        LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLFFFFFFFFF
Viterbi    LLLLLLFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls      22255544166656656356432436413151346514635341112641462625335 6
Die        FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls      366163666466232534413661661163252562462255265252266435353336
Die        LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi    LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls      233121625364414432335163243633665562466662632666612355245242
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLFFFFFFFFF
```

5’ splice site recognition

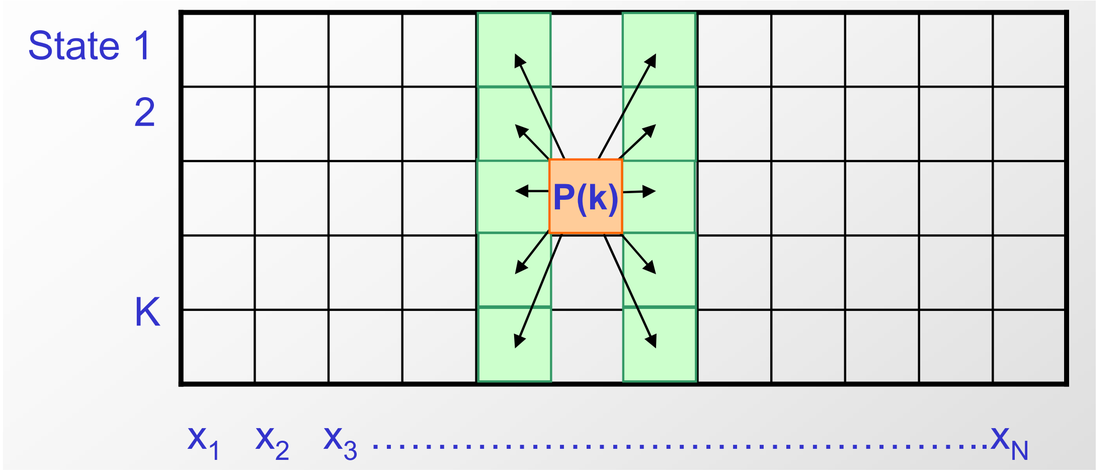


- Assumptions
  - Uniform base composition on average in exons
  - Introns are A/T rich (40% A/T, 10% G/C)
  - The 5’ splice site consensus nucleotide is almost always a G (say, 95% G and 5% A)



- What is the probability of G being a splice site given the observed sequence?

The probability of i_{th} sequence being in state k

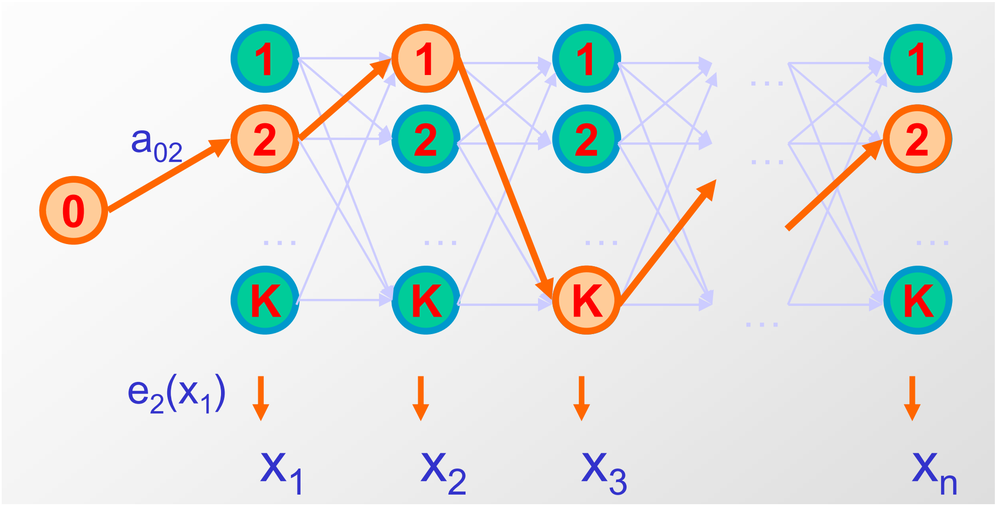


$$P(\pi_i = k|x) = \frac{P(\pi_i = k, x)}{P(x)}$$

total probability

Probability that i^{th} state is k,given all emissions x

A sequence can be generated by many different paths



How do we find the total probability of generating a given x, over all path?

Introduce the forward and backward probability

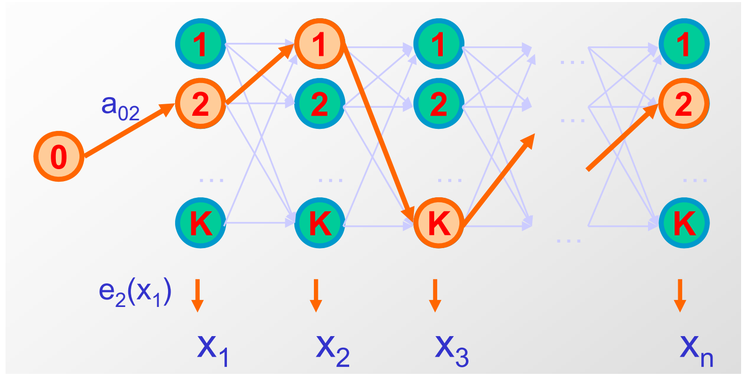
$$
\begin{aligned}
P(\pi_i = k, x) &= P(x_1 \ldots x_i, \pi_i = k, x_{i+1} \ldots x_N) \\
&= P(x_1 \ldots x_i, \pi_i = k) P(x_{i+1} \ldots x_N | x_1 \ldots x_i, \pi_i = k) \\
&= P(x_1 \ldots x_i, \pi_i = k) P(x_{i+1} \ldots x_N | \pi_i = k)
\end{aligned}
$$

forward probability

backward probability

$$= f_k(i) b_k(i)$$

The forward probability



Define the forward probability:

$$f_l(i) = P(x_1 \ldots x_i, \pi_i = l)$$

$$= \sum_{\pi_1 \ldots \pi_{i-1}} P(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1}, \pi_i = l) e_l(x_i)$$

emission probability at *i* being state *l*

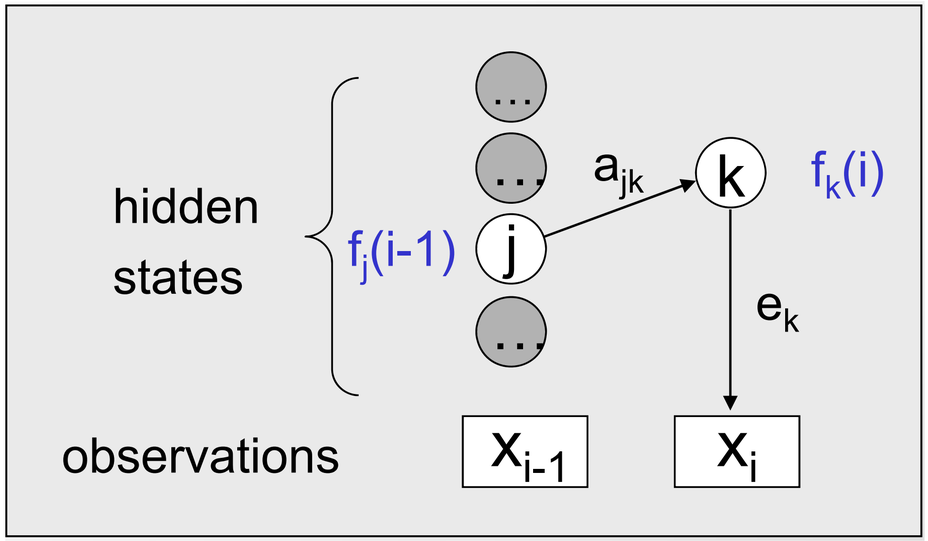
$$= \sum_k \sum_{\pi_1 \ldots \pi_{i-2}} P(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1} = k) a_{kl} e_l(x_i)$$

$$= \sum_k f_k(i-1) a_{kl} e_l(x_i)$$

total probability at *i-1* being state *k*

$$= e_l(x_i) \sum_k f_k(i-1) a_{kl}$$

Calculate the forward probability by dynamic programming



Assume we know f_j for the previous time step (i-1)

$$f_k(i) = e_k(x_i) \sum_j f_j(i-1) a_{jk}$$

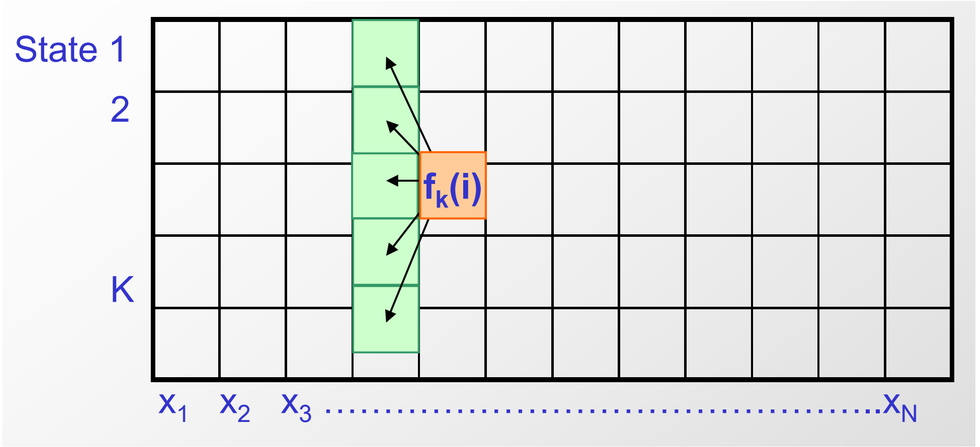
updated sum

this emission

sum ending in state j at step i-1

transition from state j

The forward algorithm



Input: x = x1……xN

Initialization:

$f_0(0)=1$, $f_k(0) = 0$, for all k > 0

Iteration:

$f_k(i) = e_K(x_i) \times \text{sum}_j\ a_{jk}\ f_j(i\text{-}1)$

Termination:

$P(x, \pi^*) = \text{sum}_k\ f_k(N)$

The forward algorithm

	Box 1	Box 2	Box 3
Red	5	4	7
White	5	6	3

Rules: The first ball is chosen from box 1, 2, 3 according to the start probability, then the remaining balls are chosen following the transition probability matrix and the emission probability matrix. Note: every time a ball is sampled, it will be put back to the box after its color is recorded.

$$S = \begin{vmatrix} 0.2 \\ 0.4 \\ 0.4 \end{vmatrix} \quad Q = \begin{pmatrix} Box\ 1 \\ Box\ 2 \\ Box\ 3 \end{pmatrix} \quad A = \begin{vmatrix} 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{vmatrix} \quad E = \begin{vmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{vmatrix}$$

$$\Sigma = \begin{pmatrix} Red \\ White \end{pmatrix}$$

What is the probability of observing “red, white, and red” balls?

$$f_0(0) = 1; f_k(0) = 0;$$

$$f_k(i) = e_k(x_i) \times \sum_j a_{jk} \cdot f_j(i-1)$$

$$i = 1: \ x_1 = Red$$

$$f_{k=1}(1) = 0.5 \cdot (0.2 \cdot 1) = 0.1$$

$$f_{k=2}(1) = 0.4 \cdot (0.4 \cdot 1) = 0.16$$

$$f_{k=3}(1) = 0.7 \cdot (0.4 \cdot 1) = 0.28$$

$$A = \begin{vmatrix} 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{vmatrix} \quad E = \begin{vmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{vmatrix}$$

$$i = 2: \ x_2 = White$$

$$f_{k=1}(2) = 0.5 \cdot (0.5 \cdot 0.1 + 0.3 \cdot 0.16 + 0.2 \cdot 0.28) = 0.077$$

$$f_{k=2}(2) = 0.6 \cdot (0.2 \cdot 0.1 + 0.5 \cdot 0.16 + 0.3 \cdot 0.28) = 0.1104$$

$$f_{k=3}(2) = 0.3 \cdot (0.3 \cdot 0.1 + 0.2 \cdot 0.16 + 0.5 \cdot 0.28) = 0.0606$$

$$i = 3: \ x_3 = Red$$

$$f_{k=1}(3) = 0.5 \cdot (0.5 \cdot 0.077 + 0.3 \cdot 0.1104 + 0.2 \cdot 0.0606) = 0.04187$$

$$f_{k=2}(3) = 0.03551$$

$$f_{k=3}(3) = 0.05284$$

$$P(x, \pi) = \sum_k f_k(n) = 0.04187 + 0.03551 + 0.05284 = 0.13022$$

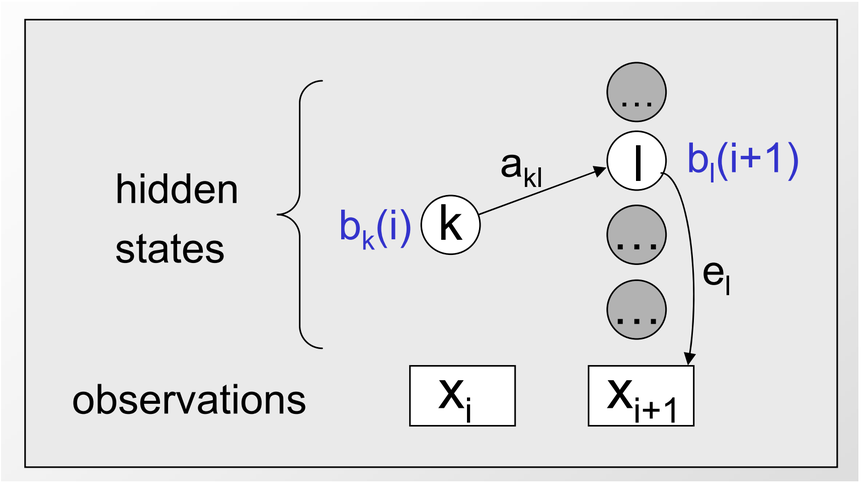
- Using the forward algorithm, we can compute the total probability of a sequence given all possible path generated by a HMM, i.e., P(x_1,…, x_n)

- Given the sequence and the HMM, what is the probability of the i^{th} sequence being in state k?

The backward probability

$$
\begin{aligned}
b_k(i) &= P(x_{i+1}...x_N|\pi_i = k) \\
&= \sum_{\pi_{i+1}...\pi_N} P(x_{i+1}x_{i+2}...x_N, \pi_{i+1}...\pi_N|\pi_i = k) \\
&= \sum_l \sum_{\pi_{i+1}...\pi_N} P(x_{i+1}x_{i+2}...x_N, \pi_{i+1} = l, \pi_{i+2}...\pi_N|\pi_i = k) \\
&= \sum_l e_l(x_{i+1})a_{kl} \sum_{\pi_{i+1}...\pi_N} P(x_{i+2}...x_N, \pi_{i+2}...\pi_N|\pi_{i+1} = l) \\
&= \sum_l e_l(x_{i+1})a_{kl}b_l(i+1)
\end{aligned}
$$

The backward probability



Assume we know b_l for the next time step *(i+1)*

$$b_k(i) = \sum_l (e_l(x_{i+1}) \times a_{kl} \times b_l(i+1))$$

current sum

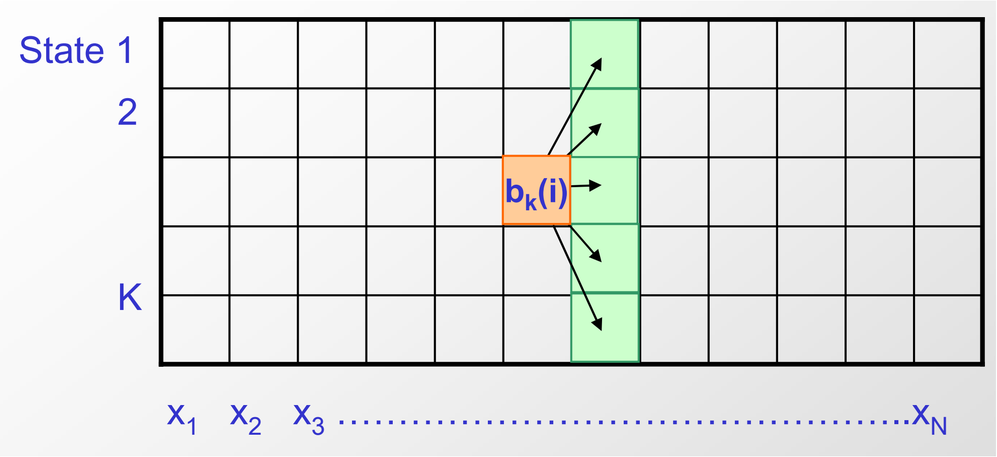
next emission

transition to state l

sum from state l to end

sum over all possible next states

The backward probability



Input: x = x1……xN

Initialization:

$b_k(N) = a_{k0}$, for all k

Iteration:

$b_k(i) = \Sigma_l\ e_l(x_{i+1})\ a_{kl}\ b_l(i+1)$

Termination:

$P(x) = \Sigma_l\ a_{0l}\ e_l(x_1)\ b_l(1)$

The backward algorithm

	Box 1	Box 2	Box 3
Red	5	4	7
White	5	6	3

Rules: The first ball is chosen from box 1, 2, 3 according to the start probability, then the remaining balls are chosen following the transition probability matrix and the emission probability matrix. Note: every time a ball is sampled, it will be put back to the box after its color is recorded.

$$S = \begin{vmatrix} 0.2 \\ 0.4 \\ 0.4 \end{vmatrix} \quad Q = \begin{pmatrix} Box\ 1 \\ Box\ 2 \\ Box\ 3 \end{pmatrix} \quad A = \begin{vmatrix} 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{vmatrix} \quad E = \begin{vmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{vmatrix}$$

$$\Sigma = \begin{pmatrix} Red \\ White \end{pmatrix}$$

What is the probability of observing “red, white, and red” balls?

$$b_k(N) = A_{k0} = 1$$

$$b_k(i) = \sum_l e_l(x_{i+1}) \times a_{kl} \cdot b_l(i+1)$$

$$A = \begin{vmatrix} 0.5 & 0.2 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{vmatrix} \quad E = \begin{vmatrix} 0.5 & 0.5 \\ 0.4 & 0.6 \\ 0.7 & 0.3 \end{vmatrix}$$

$i = 3: \;\; x_3 = Red$

$$b_{k=1}(3) = 1; \;\; b_{k=2}(3) = 1; \;\; b_{k=3}(3) = 1$$

$i = 2: \;\; x_2 = White$

$$b_{k=1}(2) = (0.5 \cdot 0.5 \cdot 1 + 0.4 \cdot 0.2 \cdot 1 + 0.7 \cdot 0.3 \cdot 1) = 0.54$$

$$b_{k=2}(2) = (0.5 \cdot 0.3 \cdot 1 + 0.4 \cdot 0.5 \cdot 1 + 0.7 \cdot 0.2 \cdot 1) = 0.49$$

$$b_{k=3}(2) = (0.5 \cdot 0.2 \cdot 1 + 0.4 \cdot 0.3 \cdot 1 + 0.7 \cdot 0.5 \cdot 1) = 0.57$$

$i = 1: \;\; x_1 = Red$

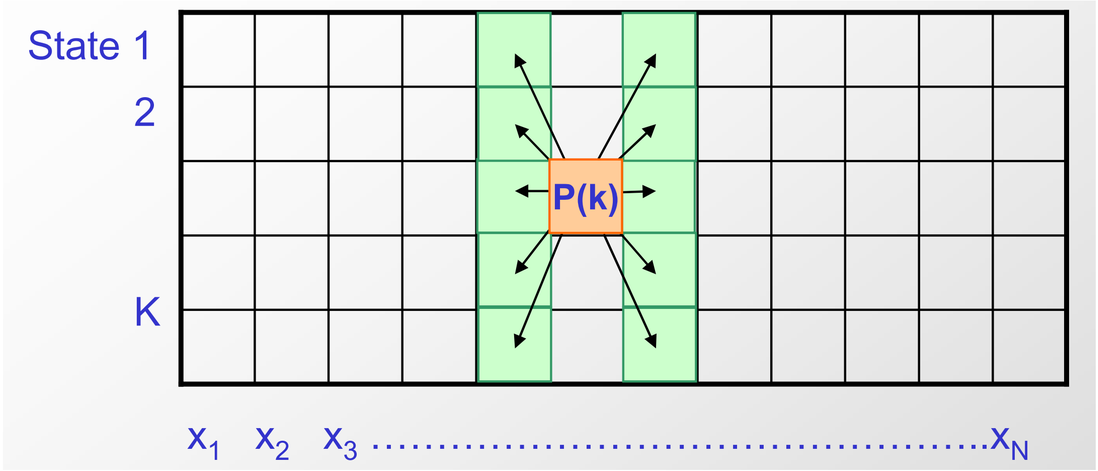
$$b_{k=1}(1) = (0.5 \cdot 0.5 \cdot 0.54 + 0.6 \cdot 0.2 \cdot 0.49 + 0.3 \cdot 0.3 \cdot 0.57) = 0.2451$$

$$b_{k=2}(1) = (0.5 \cdot 0.3 \cdot 0.54 + 0.6 \cdot 0.5 \cdot 0.49 + 0.3 \cdot 0.2 \cdot 0.57) = 0.2622$$

$$b_{k=3}(1) = (0.5 \cdot 0.2 \cdot 0.54 + 0.6 \cdot 0.3 \cdot 0.49 + 0.3 \cdot 0.5 \cdot 0.57) = 0.2277$$

$$P(x, \pi) = (0.5 \cdot 0.2 \cdot 0.2451 + 0.4 \cdot 0.4 \cdot 0.2622 + 0.7 \cdot 0.4 \cdot 0.2277) = 0.13022$$

The forward-backword algorithm for posterior decoding



$$P(\pi_i = k|x) = \frac{P(\pi = k, x)}{P(X)} = \frac{f_k(i) \times b_k(i)}{P(x)}$$

Probability that i^{th} state is k,given all emissions x

The probability of the ith state being K

	Box 1	Box 2	Box 3
Red	5	4	7
White	5	6	3

Given the observation of “red, white, red”, what is the probability of “white” being from Box1, 2, and 3?

$$P(\pi_i = k|x) = \frac{P(\pi = k, x)}{P(X)} = \frac{f_k(i) \times b_k(i)}{P(x)}$$

$$P(x, \pi) = \sum_k f_k(n) = 0.04187 + 0.03551 + 0.05284 = 0.13022$$

$$f_{k=1}(2) = 0.5 \cdot (0.5 \cdot 0.1 + 0.3 \cdot 0.16 + 0.2 \cdot 0.28) = 0.077$$

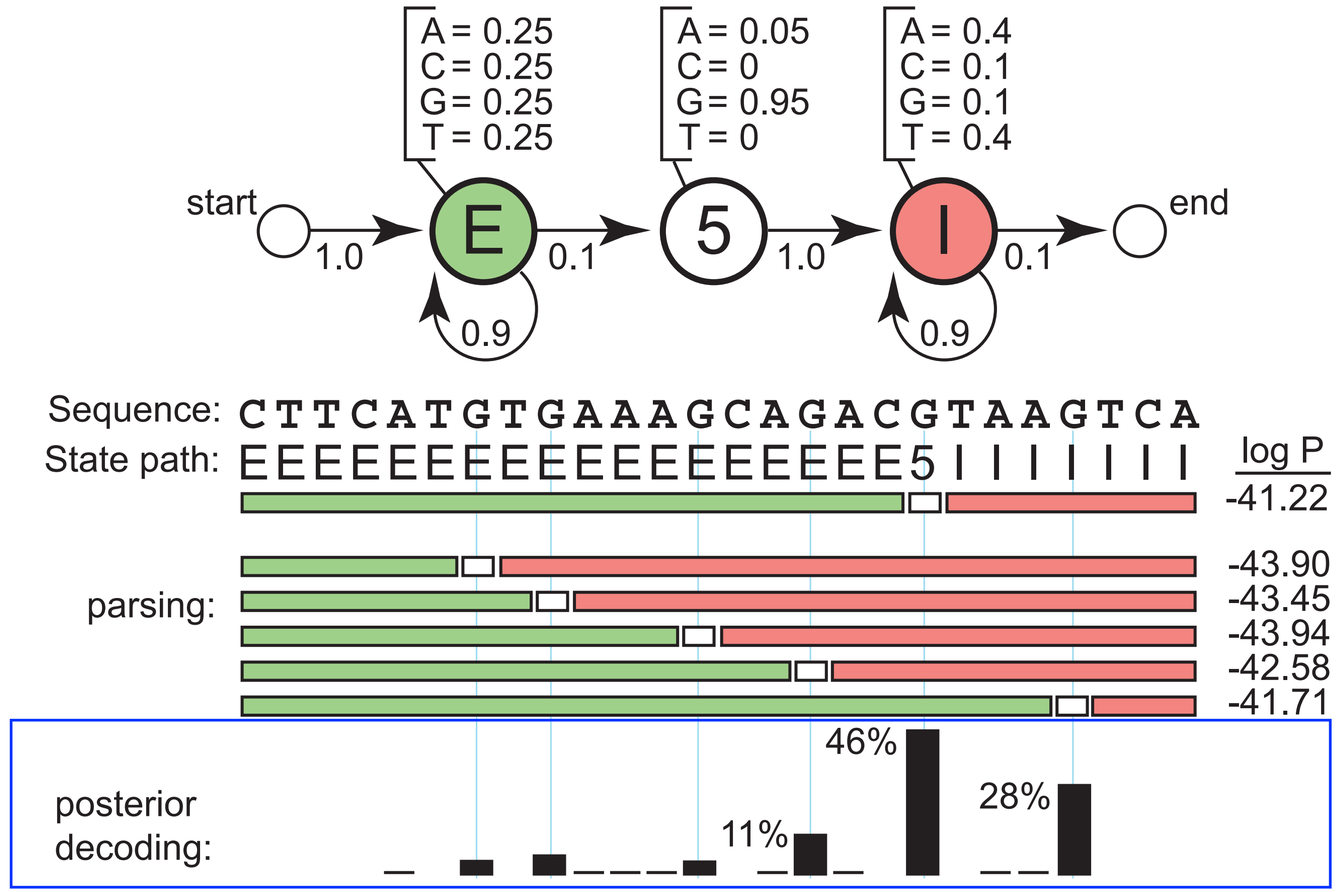
$$b_{k=1}(2) = (0.5 \cdot 0.5 \cdot 1 + 0.4 \cdot 0.2 \cdot 1 + 0.7 \cdot 0.3 \cdot 1) = 0.54$$

$$P(\pi_2 = box\ 1|x) = \frac{f_1(2) \times b_1(2)}{P(x)} = \frac{0.077 \cdot 0.54}{0.13022} = 0.3193$$

$$P(\pi_2 = box\ 2|x) = \frac{f_2(2) \times b_2(2)}{P(x)} = \frac{0.1104 \cdot 0.49}{0.13022} = 0.4154$$

$$P(\pi_2 = box\ 3|x) = \frac{f_3(2) \times b_3(2)}{P(x)} = \frac{0.0606 \cdot 0.57}{0.13022} = 0.2653$$

The 5’ splice site recognition



- What if M is unknown, and we are given a set of sequences {x^1, …, x^n}.

- How can we estimate the parameters in HMM (transition probabilities and emission probabilities)?

Let Θ be a vector combining the unknown transition and emission probabilities of the HMM $\mathcal{M}$. Given an observed symbol string x that the HMM emitted, define $P(x|\Theta)$ as the maximum probability of x given the assignment of parameters Θ. Our goal is to find

$$\max_{\Theta} P(x|\Theta).$$

Usually, instead of a single string x, we can obtain a sample of *training sequences* $x^1, \ldots, x^m$, so a natural goal is to find

$$\max_{\Theta} \prod_{j=1}^{m} P(x^j|\Theta).$$

The states in each training sequence are known

If A_{kl} is the number of transitions from state k to l and $E_k(b)$ is the number of times b is emitted from state k, then the reasonable estimators are:

$$a_{kl} = \frac{A_{kl}}{\sum_{q \in Q} A_{kq}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{\sigma \in \sum} E_k(\sigma)}.$$

A_{kl} = number of transitions k to l in training data + r_{kl},

Pseudocounts

$E_k(b)$ = number of emissions of b from k in training data + $r_k(b)$.

The states in each training sequence are unknown

The probability a_{kl} is used in i^{th} position in sequence x is:

$$P(\pi_i = k, \pi_{i+1} = l | x, \theta) = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(x)}.$$

forward probability

backword probability

The expected number of A_{kl} can be calculated as:

$$A_{kl} = \sum_j \frac{1}{P(x^j)} \sum_i f_k^j(i) a_{kl} e_l(x_{i+1}^j) b_l^j(i+1),$$

The expected number of $E_k(b)$ can be calculated as:

$$E_k(b) = \sum_j \frac{1}{P(x^j)} \sum_{\{i | x_i^j = b\}} f_k^j(i) b_k^j(i),$$

The Baum-Welch Algorithm

Initialization:

Pick the best-guess for model parameters

(or arbitrary)

Iteration:

1. Forward
2. Backward
3. Calculate A_{kl}, $E_k(b)$
4. Calculate new model parameters a_{kl}, $e_k(b)$
5. Calculate new log-likelihood $P(x \mid \theta)$

Until $P(x \mid \theta)$ does not change much

Package 'HMM'

February 19, 2015

Type Package

Version 1.0

Title HMM - Hidden Markov Models

Date 2010-01-10

Maintainer Lin Himmelmann <hmm@linhi.com>

Author Scientific Software Development - Dr. Lin Himmelmann, www.linhi.com

Depends R (>= 2.0.0)

Description Easy to use library to setup, apply and make inference with discrete time and discrete space Hidden Markov Models

License GPL (>= 2)

Repository CRAN

Date/Publication 2010-01-10 08:04:29

R topics documented:

Pfam 27.0 (March 2013, 14831 families)

The Pfam database is a large collection of protein families, each represented by **multiple sequence alignments** and **hidden Markov models (HMMs)**. **More...**

QUICK LINKS	YOU CAN FIND DATA IN PFAM IN VARIOUS WAYS...
SEQUENCE SEARCH	Analyze your protein sequence for Pfam matches
VIEW A PFAM FAMILY	View Pfam family annotation and alignments
VIEW A CLAN	See groups of related families
VIEW A SEQUENCE	Look at the domain organisation of a protein sequence
VIEW A STRUCTURE	Find the domains on a PDB structure
KEYWORD SEARCH	Query Pfam by keywords
JUMP TO	enter any accession or ID Go Example

Enter any type of accession or ID to jump to the page for a Pfam family or clan, UniProt sequence, PDB structure, etc.

Or view the help pages for more information

Recent Pfam blog posts

☒Hide this

Short-term Pfam position available. (posted 7 February 2014)

We have just advertised a 9-month maternity cover position in Pfam. We are looking for a skilled

HMMER

biosequence analysis using profile hidden Markov models

HHMI janelia farm research campus

Home | Search | Results | Software | Help | About

Search

Upload a sequence, HMM or alignment and perform a HMMER search against one of seven sequence databases or the Pfam HMM database.

Start

hmmsearch jackhmmer hmmscan phmmer

Documentation

Download the documentation for the **command line** version of HMMER. (PDF, 478 KB)

Read the online help for the HMMER **webserver** search service.

Download HMMER

Get the latest version

v3.1b1

Download (MacOSX/Intel)

Alternative Download Options

HMMER is used for searching sequence databases for homologs of protein sequences, and for making protein sequence alignments. It implements methods using probabilistic models called **profile hidden Markov models** (profile HMMs).

Compared to BLAST, FASTA, and other sequence alignment and database search tools based on older scoring methodology, HMMER aims to be significantly *more* accurate and *more* able to detect remote homologs because of the strength of its underlying mathematical models. In the past, this strength came at significant computational expense, but in the new HMMER3 project, HMMER is now essentially **as fast as** BLAST.

As part of this evolution in the HMMER software, we are committed to making the software available to as many scientists as possible. Earlier releases of HMMER were restricted to command line use. To make the software more accessible to the wide scientific community, we now provide **servers** that allow sequence searches to be performed interactively via the Web.

What is a hidden Markov model?

Sean R Eddy

Statistical models called hidden Markov models are a recurring theme in computational biology. What are hidden Markov models, and why are they so useful for so many different problems?

Often, biological sequence analysis is just a matter of putting the right label on each residue. In gene identification, we want to label nucleotides as exons, introns, or intergenic sequence. In sequence alignment, we want to associate residues in a query sequence with homologous residues in a target database sequence. We can always write an *ad hoc* program for any given problem, but the same frustrating issues will always recur. One is that we want to incorporate heterogeneous sources of information. A genefinder, for instance, ought to combine splice-site consensus, codon bias, exon/intron length preferences and open reading frame analysis into one scoring system. How should these parameters be set? How should different kinds of information be weighted? A second issue is to interpret results probabilistically. Finding a best scoring answer is one thing, but what does the score mean, and how confident are we that the best scoring answer is correct? A third issue is exten-

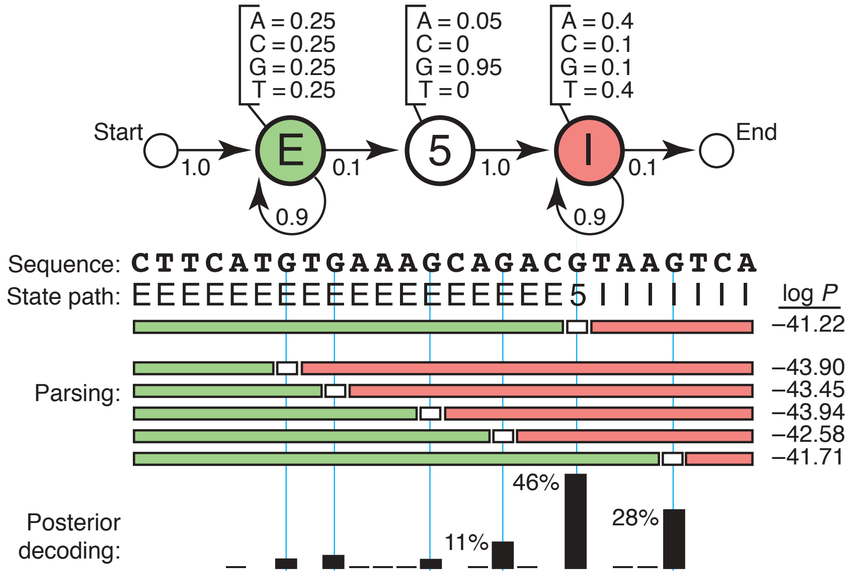


Figure 1 A toy HMM for 5′ splice site recognition. See text for explanation.