Phylogenetics:
Parsimony and Likelihood

COMP 571 - Spring 2015

Luay Nakhleh, Rice University

The Problem

- Input: Multiple alignment of a set *S* of sequences
- Output: Tree *T* leaf-labeled with *S*

Assumptions

- Characters are mutually independent
- Following a speciation event, characters continue to evolve independently

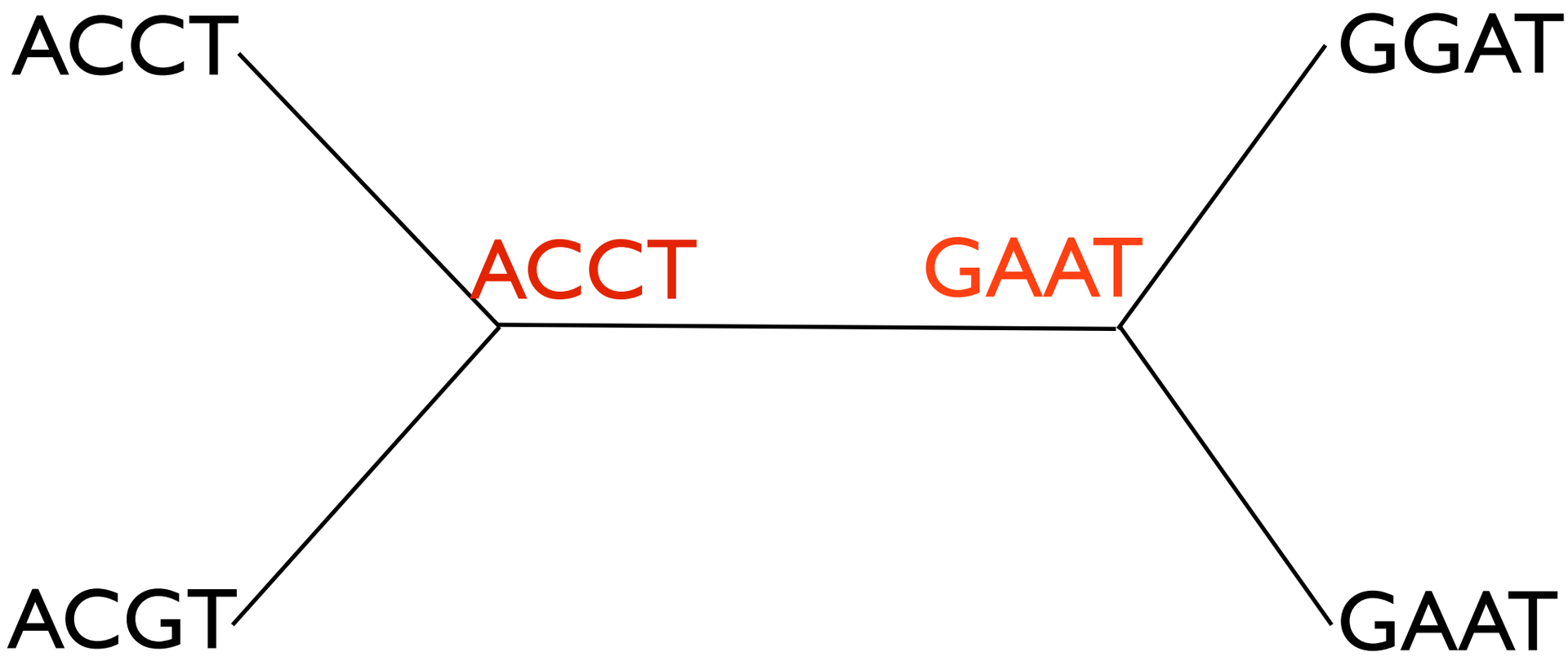
- Usually, the inferred tree (in character-based methods) is fully labeled

ACCT

GGAT

ACGT

GAAT


ACCT
GGAT
ACCT
GAAT
ACGT
GAAT

A Simple Solution: Try All Trees

- Problem:
  - (2n-3)!! rooted trees
  - (2m-5)!! unrooted trees

A Simple Solution: Try All Trees

Numbers of possible rooted and unrooted trees for up to 20 OTUs		
Number of OTUs	Number of rooted trees	Number of unrooted trees
2	1	1
3	3	1
4	15	3
5	105	15
6	945	105
7	10,395	954
8	135,135	10,395
9	2,027,025	135,135
10	34,459,425	2,027,025
11	654,729,075	34,459,425
12	13,749,310,575	654,729,075
13	316,234,143,225	13,749,310,575
14	7,905,853,580,625	316,234,143,225
15	213,458,046,676,875	7,905,853,580,625
16	6,190,283,353,629,375	213,458,046,676,875
17	191,898,783,962,510,625	6,190,283,353,629,375
18	6,332,659,870,762,850,625	191,898,783,962,510,625
19	221,643,095,476,699,771,875	6,332,659,870,762,850,625
20	8,200,794,532,637,891,559,375	221,643,095,476,699,771,875

Data from Felsenstein

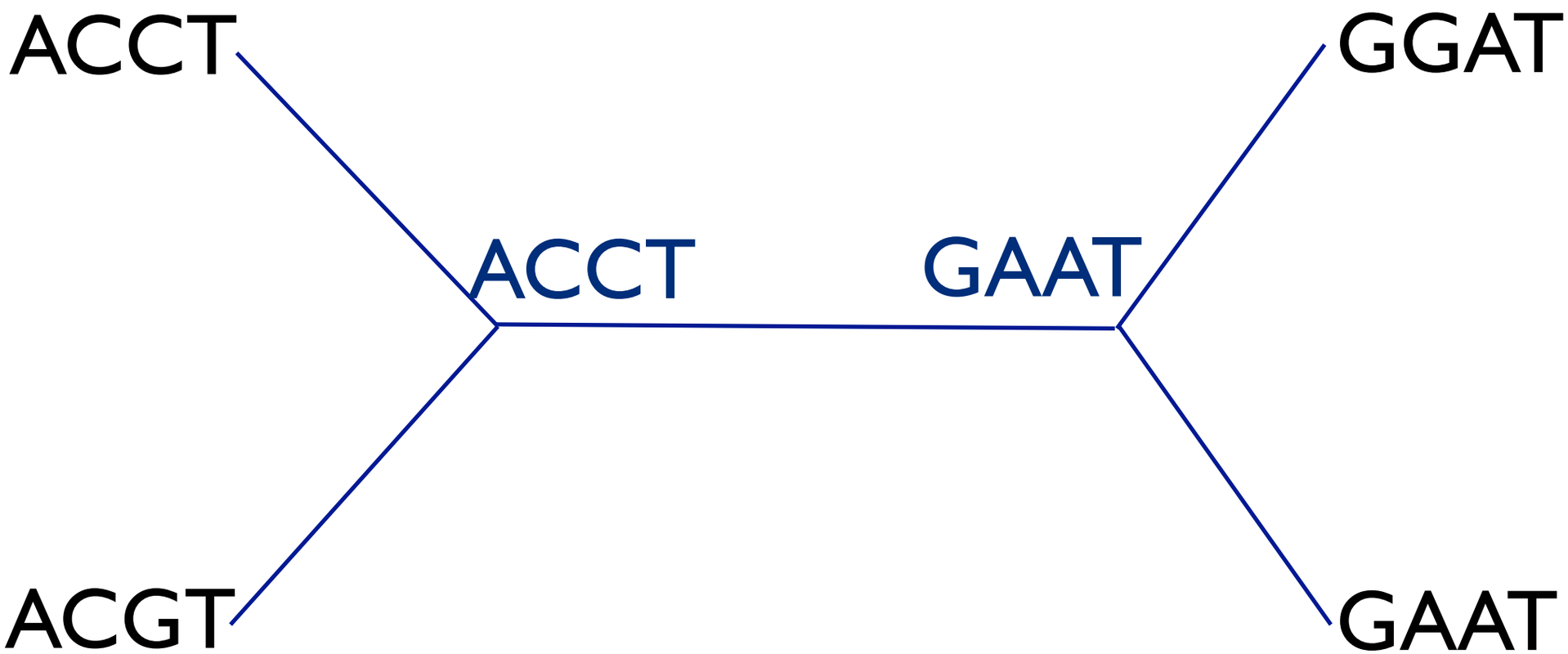
Solution

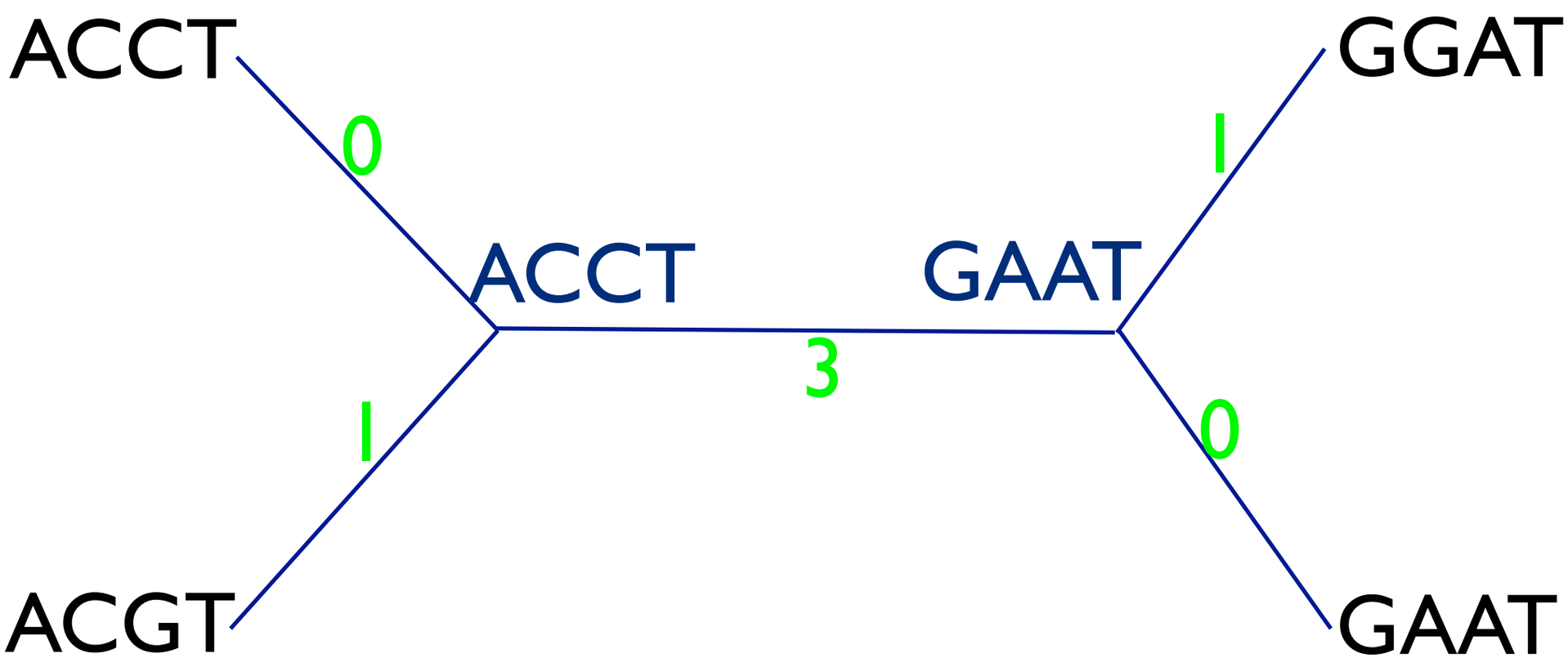
- Define an optimization criterion
- Find the tree (or, set of trees) that optimizes the criterion
- Two common criteria: parsimony and likelihood

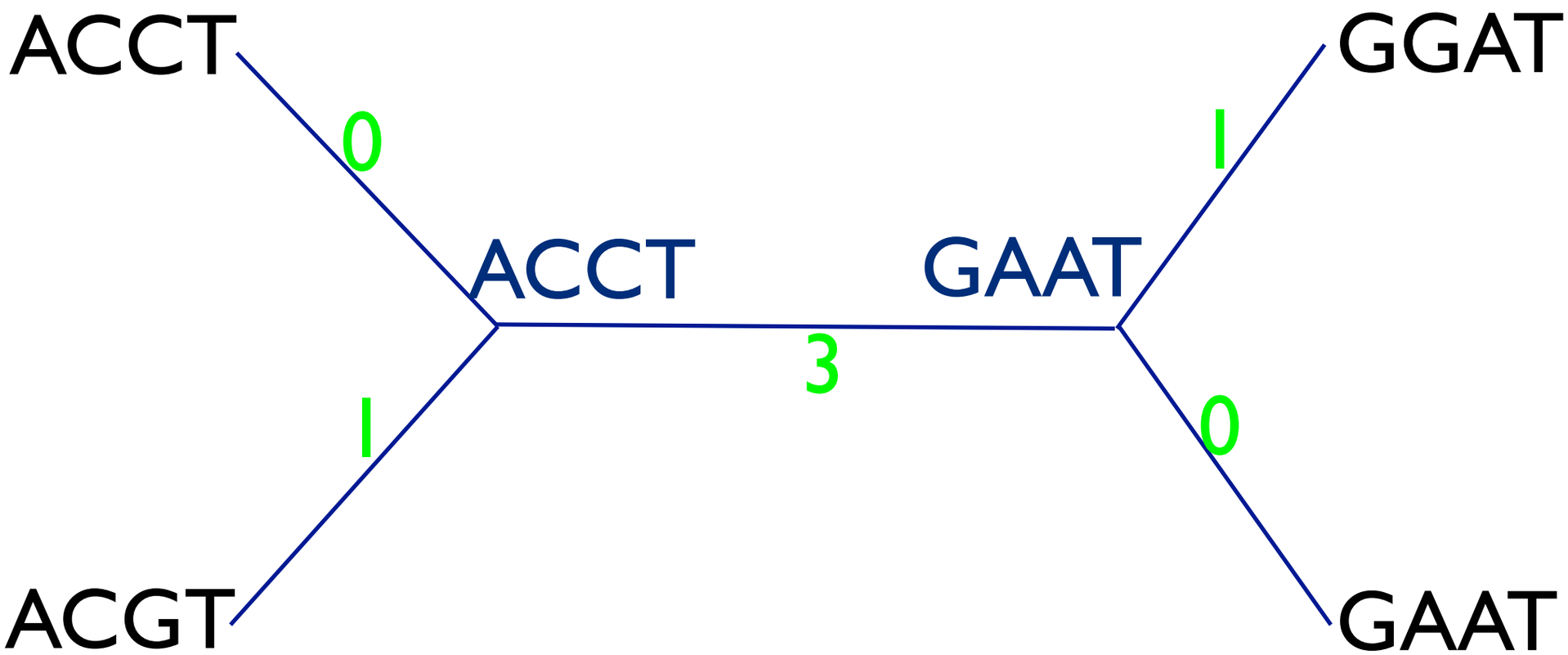
Parsimony

- The parsimony of a fully-labeled unrooted tree *T*, is the sum of lengths of all the edges in *T*
- Length of an edge is the Hamming distance between the sequences at its two endpoints

- PS(T)


ACCT
GGAT
ACCT
GAAT
ACGT
GAAT


ACCT
0
ACCT
GAAT
1
3
1
0
ACGT
GAAT

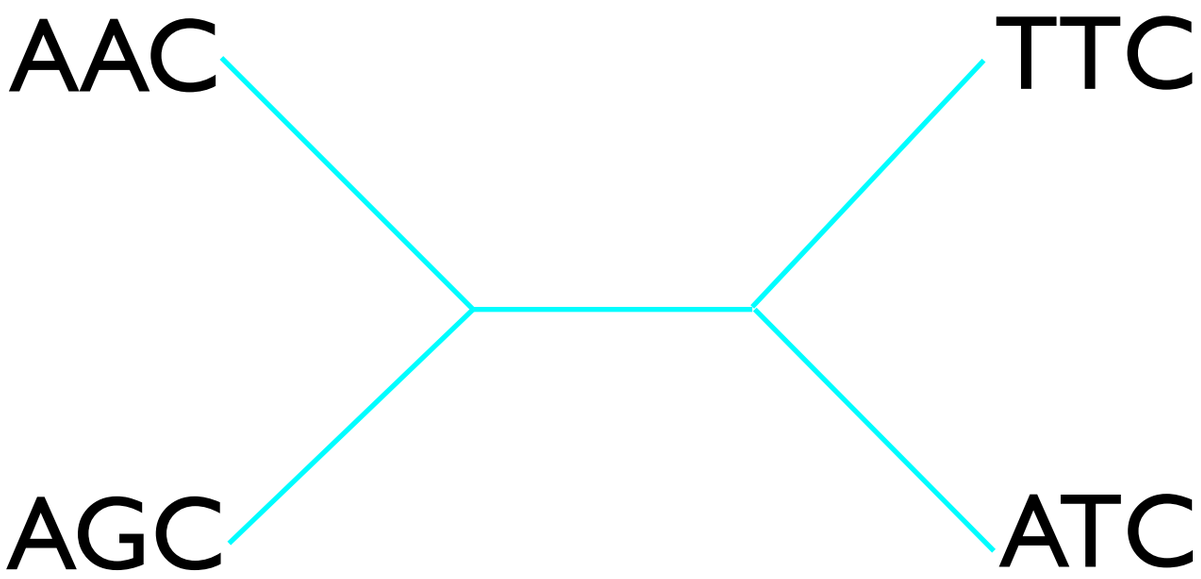


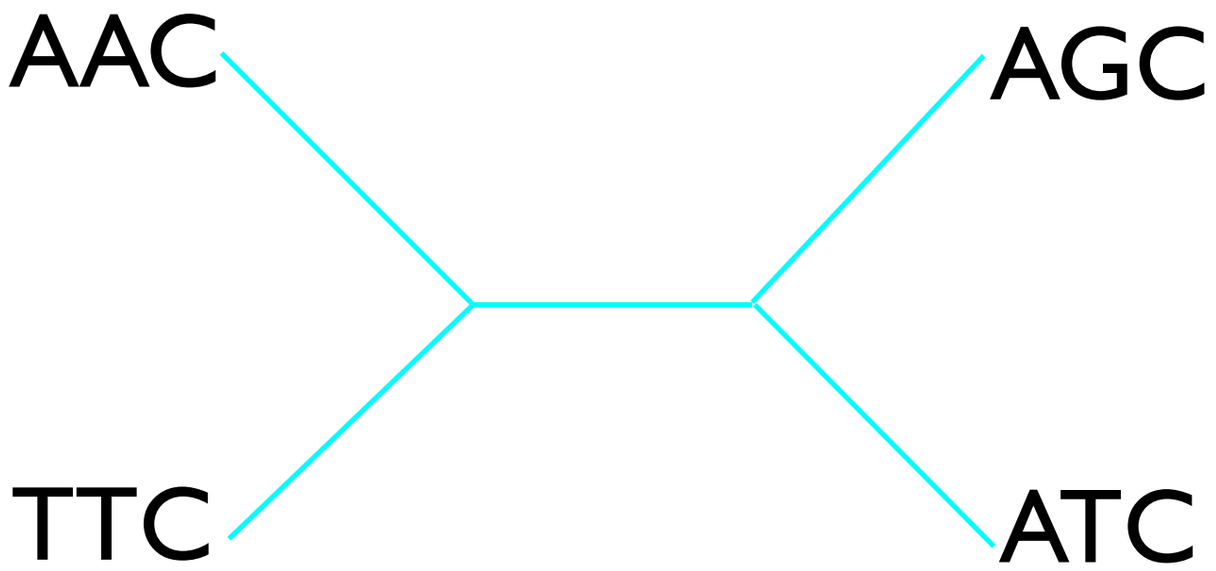
Parsimony score = 5

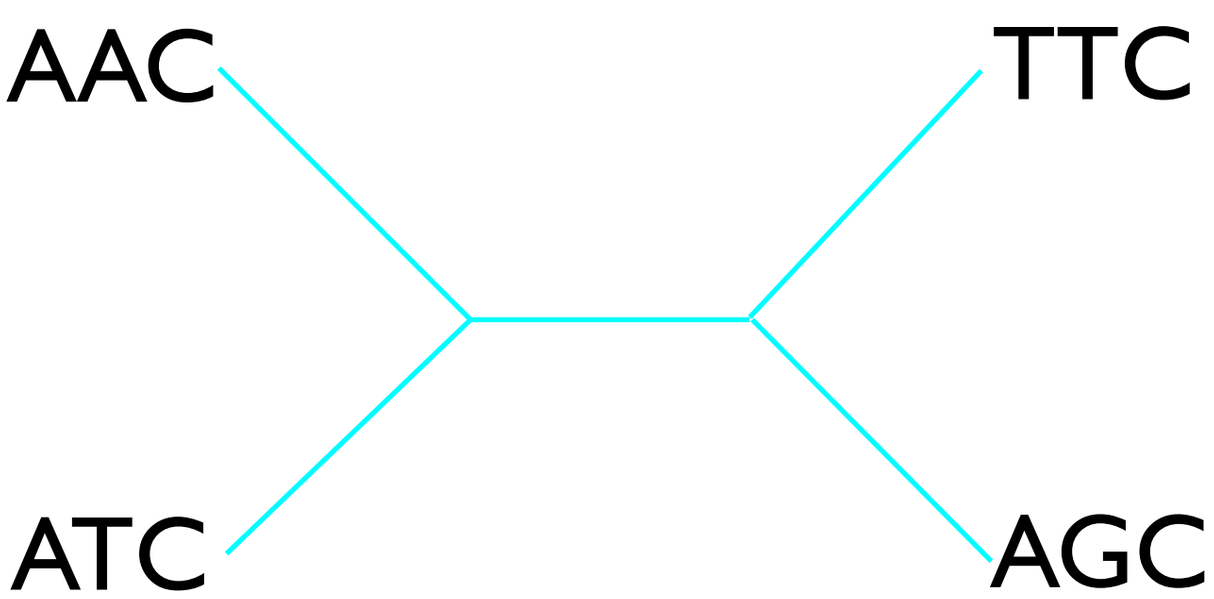
Maximum Parsimony (MP)

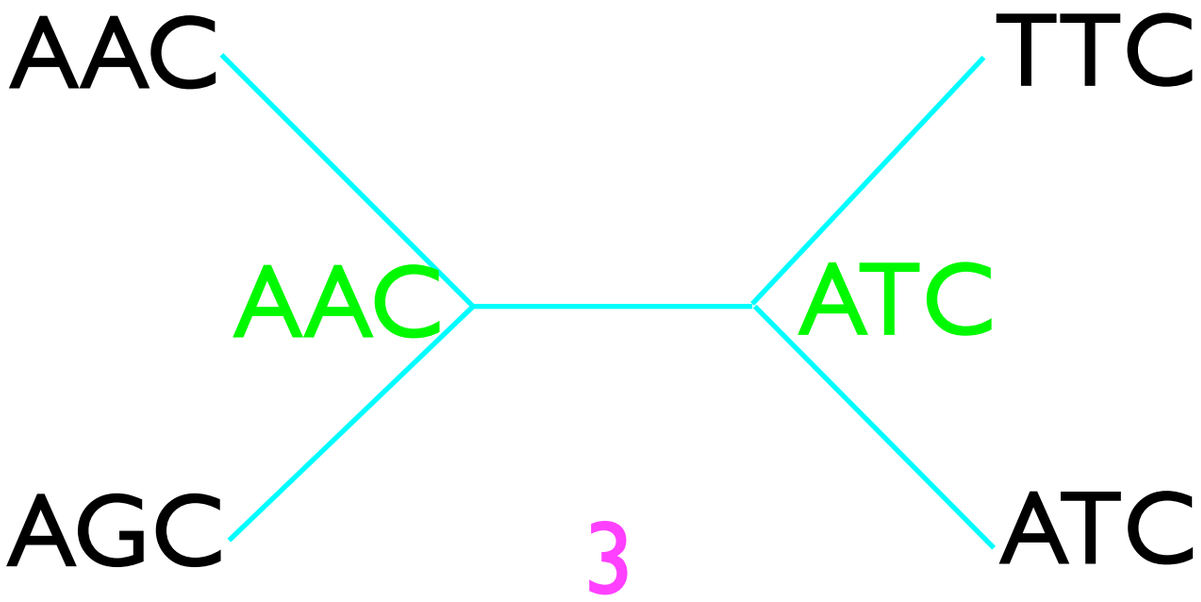
- **Input:** a multiple alignment *S* of *n* sequences
- **Output:** tree *T* with *n* leaves, each leaf labeled by a unique sequence from *S*, internal nodes labeled by sequences, and *PS(T)* is minimized

AAC AGC TTC ATC

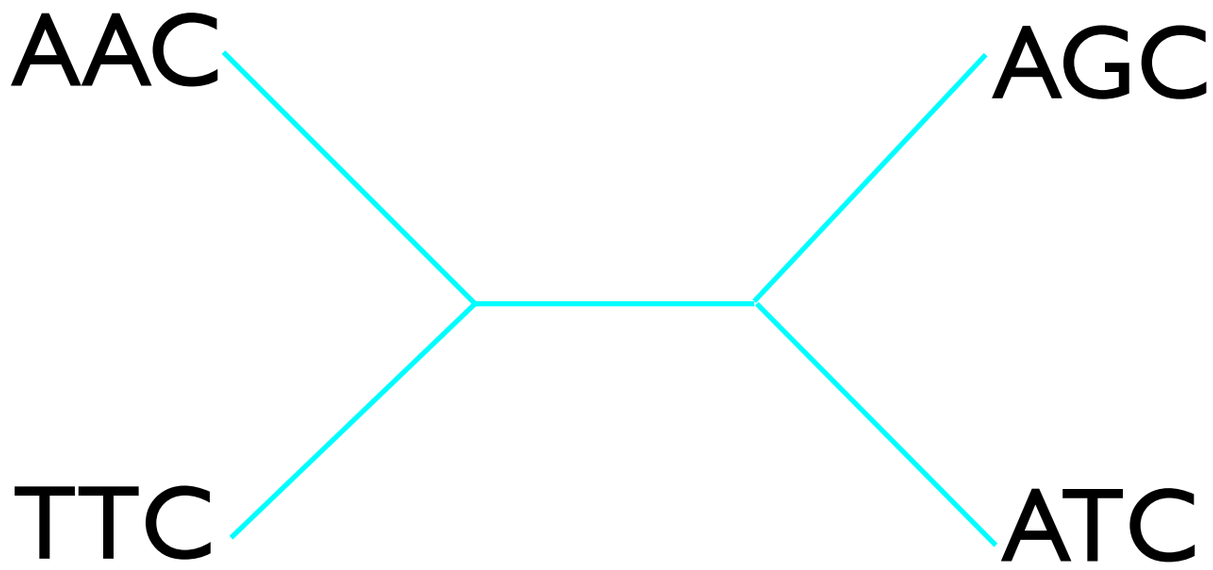

AAC
TTC
AGC
ATC

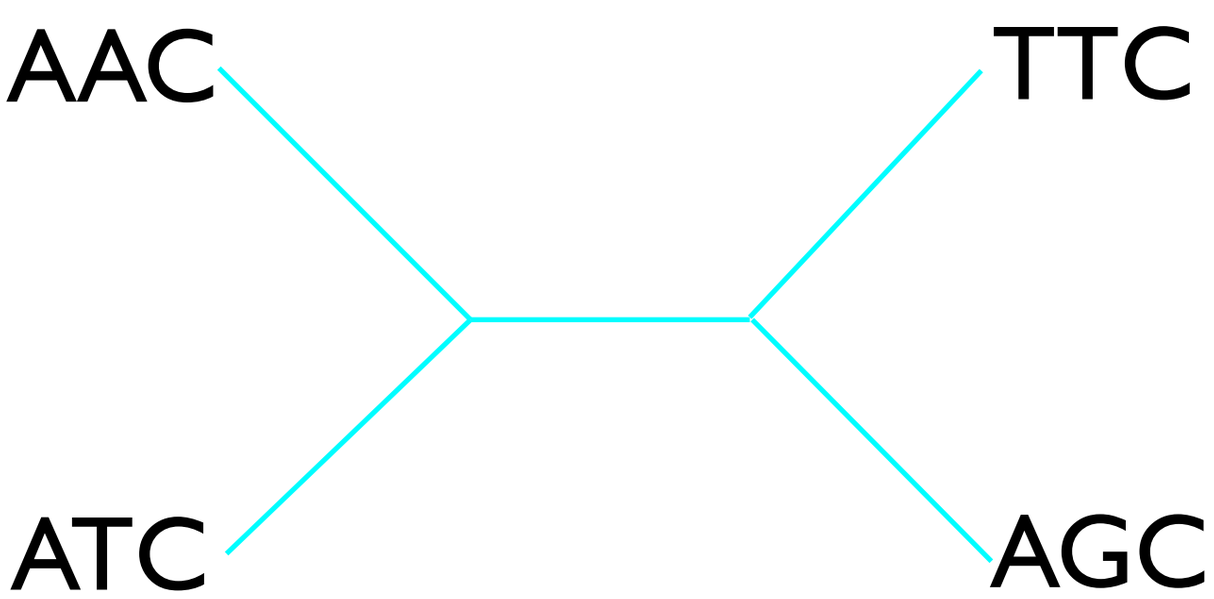

AAC
AGC
TTC
ATC

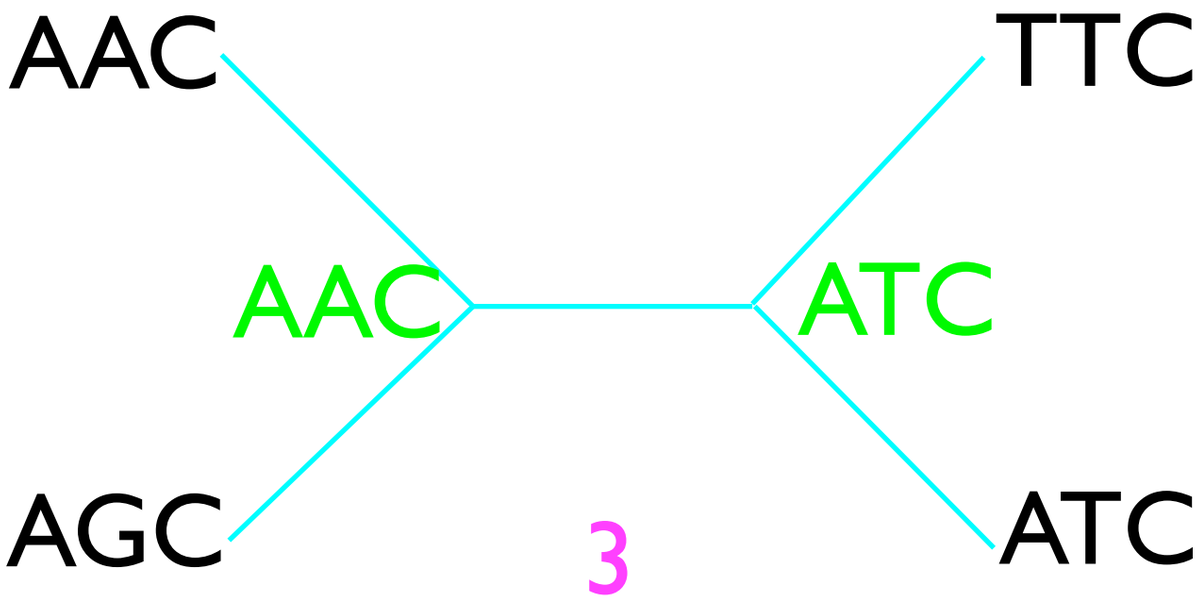

AAC
TTC
ATC
AGC

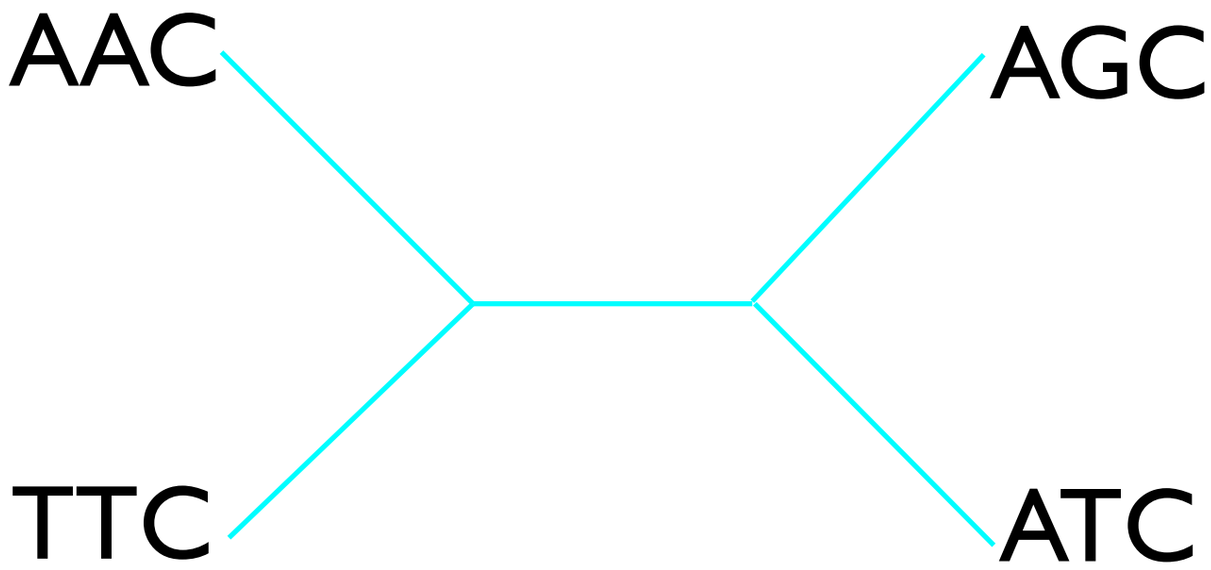


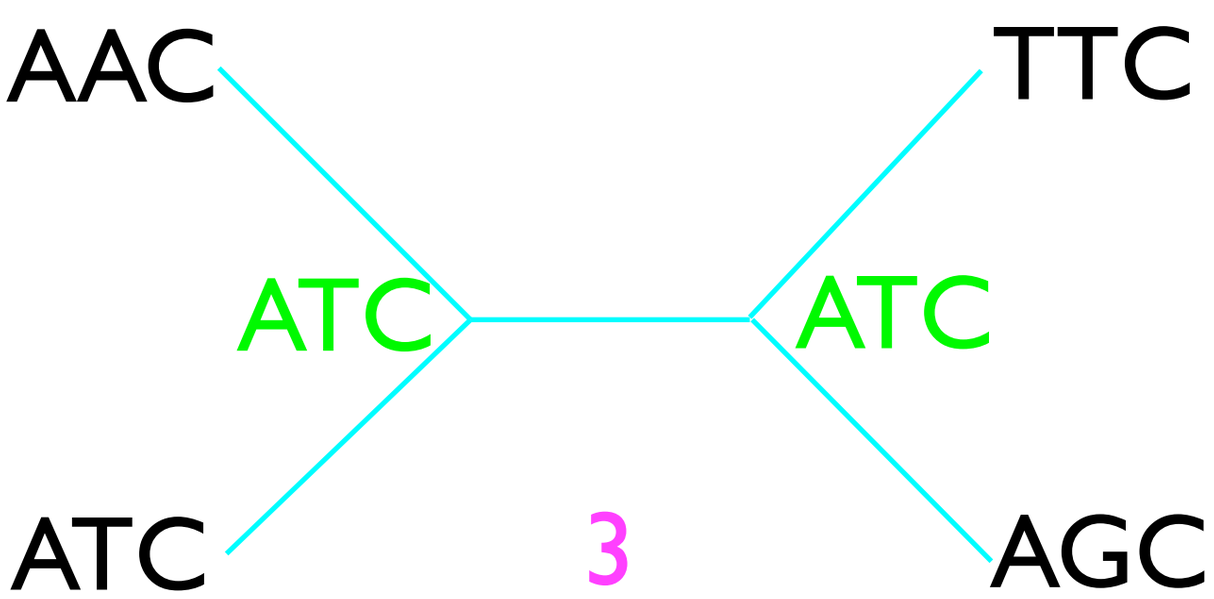
3

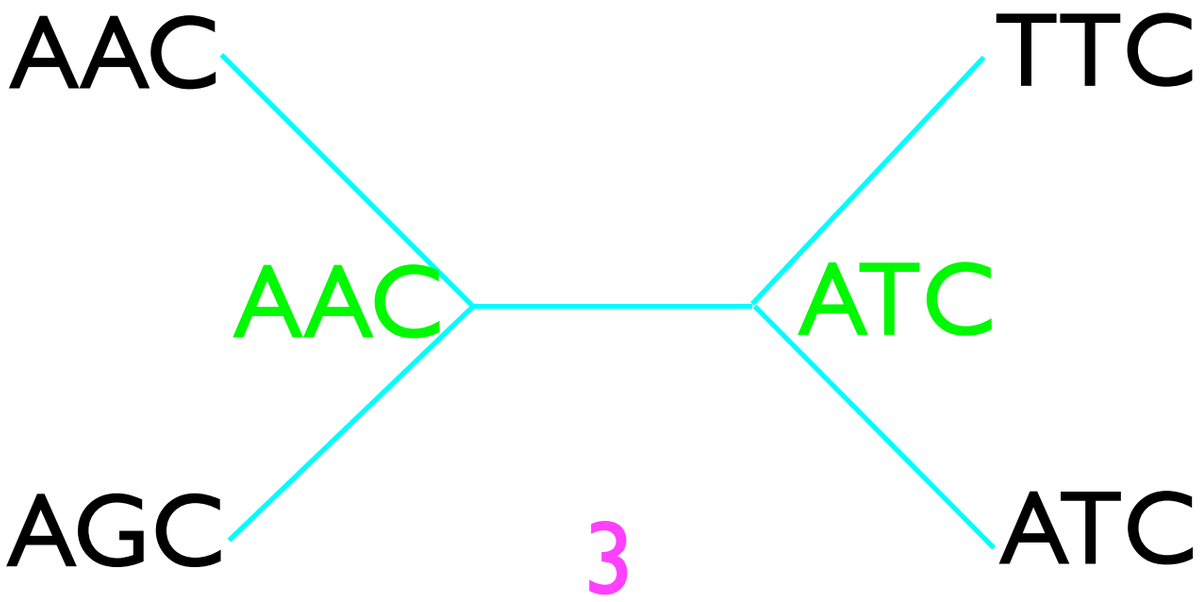


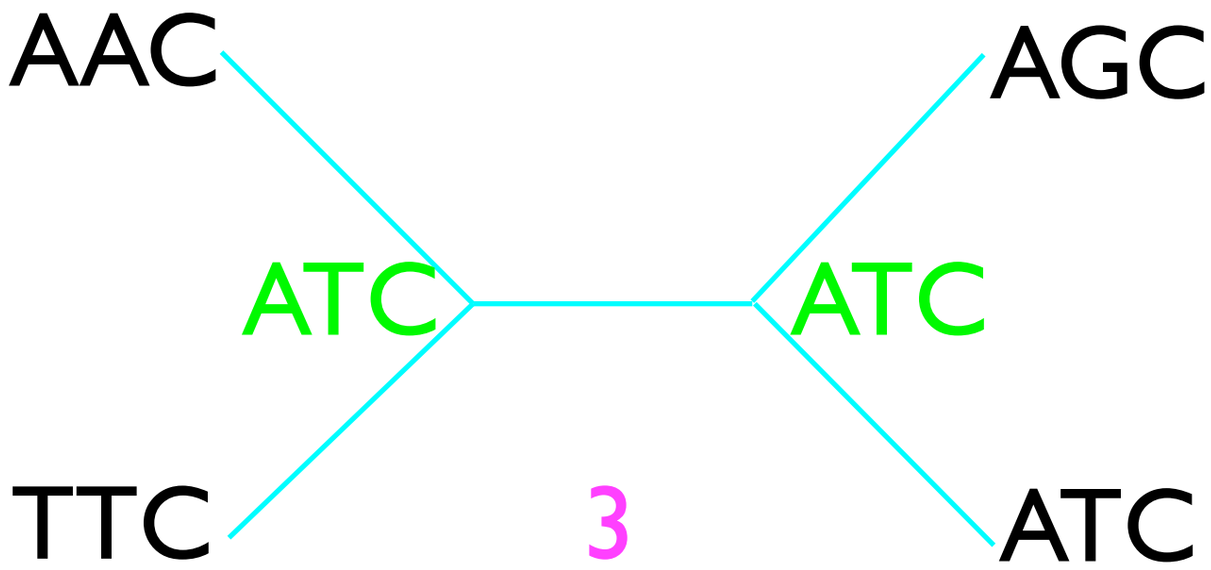


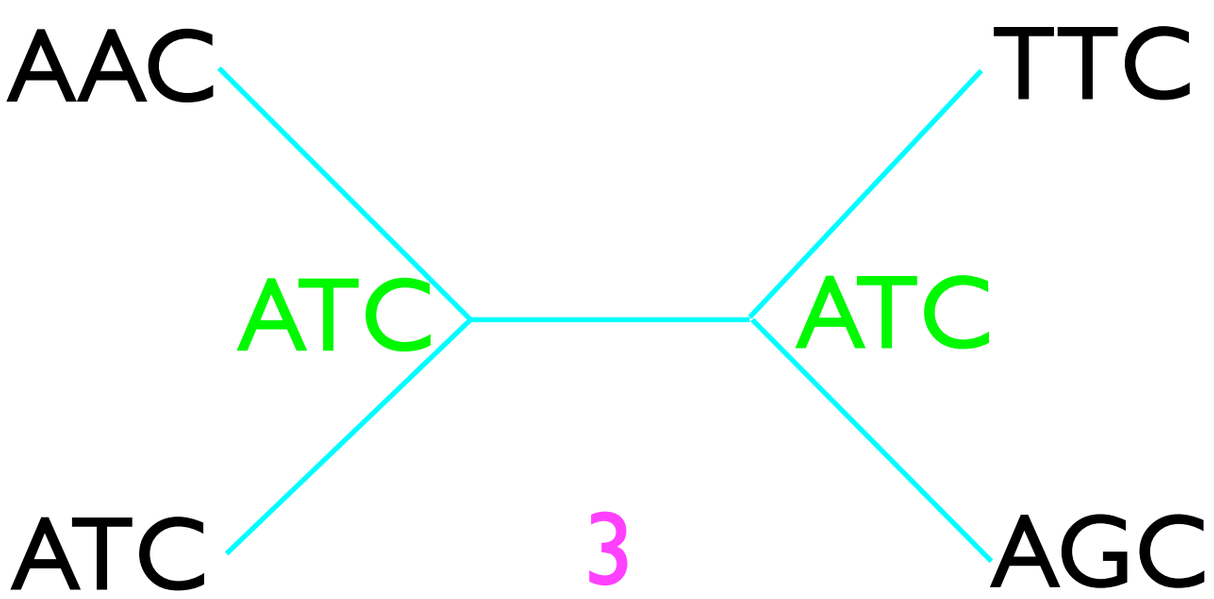

AAC
TTC
AAC
ATC
AGC
ATC
3

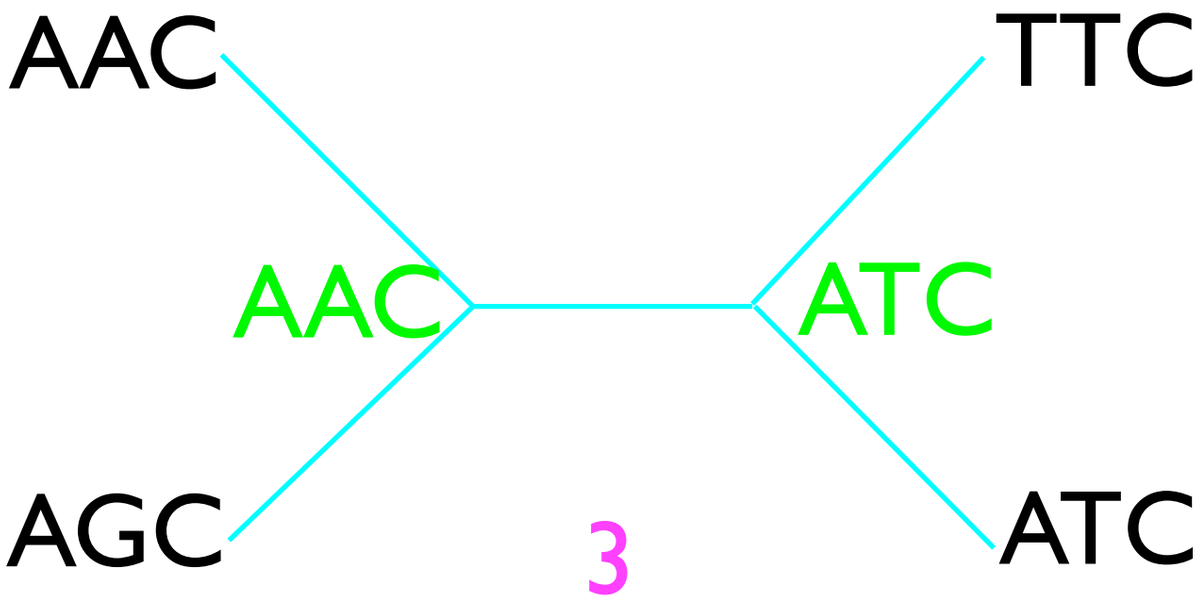

AAC
AGC
TTC
ATC


AAC
TTC
ATC
ATC
ATC
AGC
3

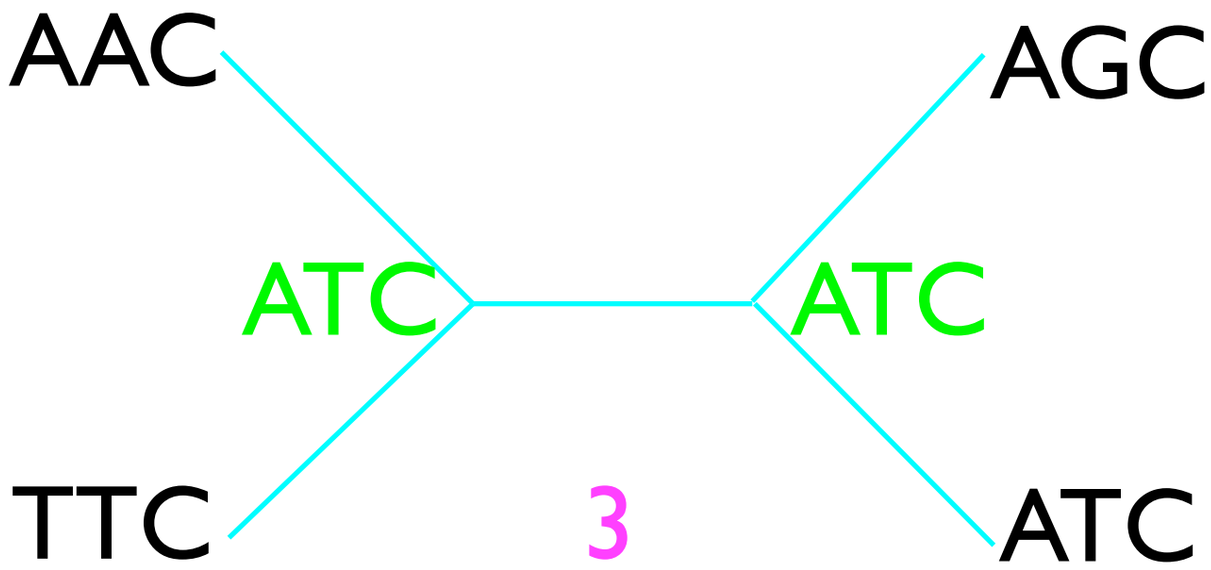

AAC
TTC
AAC
ATC
AGC
ATC
3

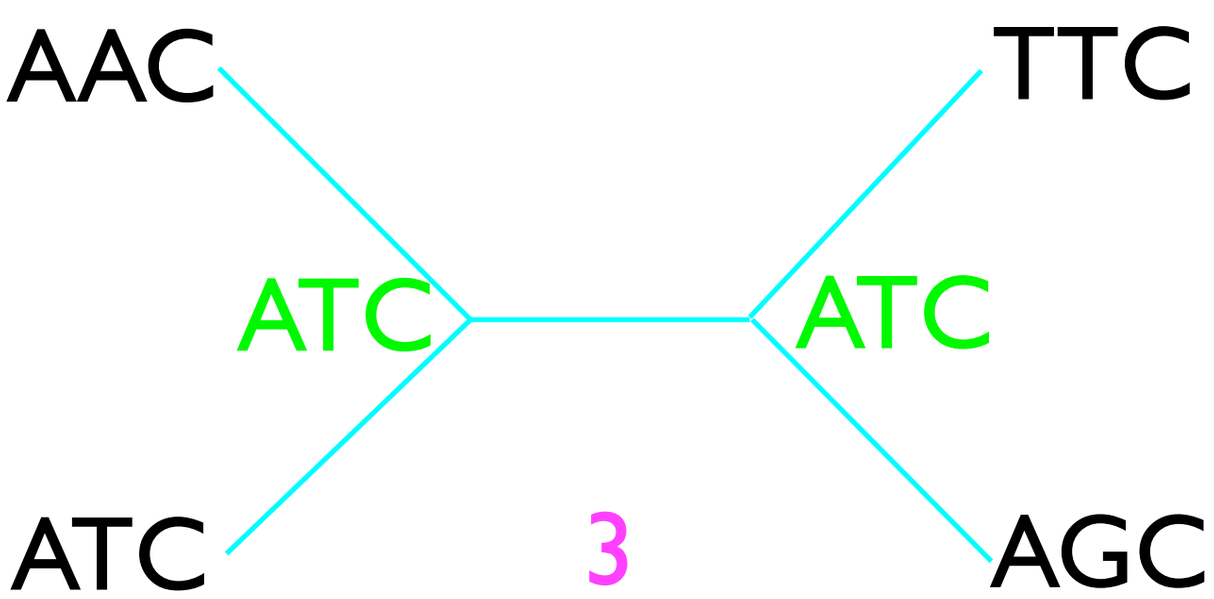

AAC
AGC
ATC
ATC
TTC
3
ATC


AAC
TTC
ATC
ATC
ATC
3
AGC

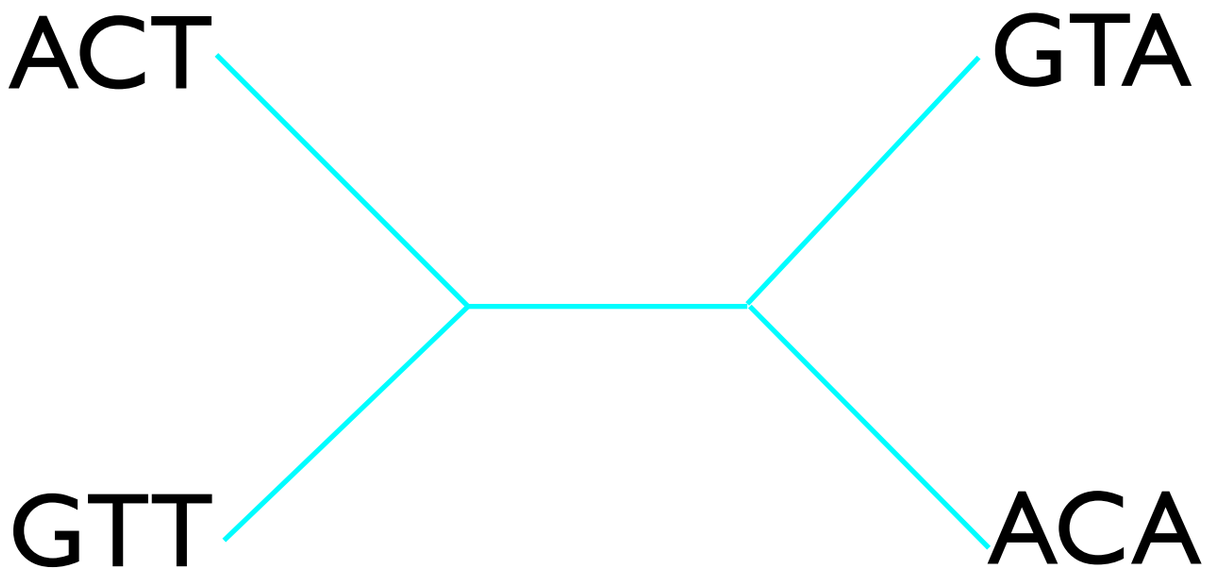


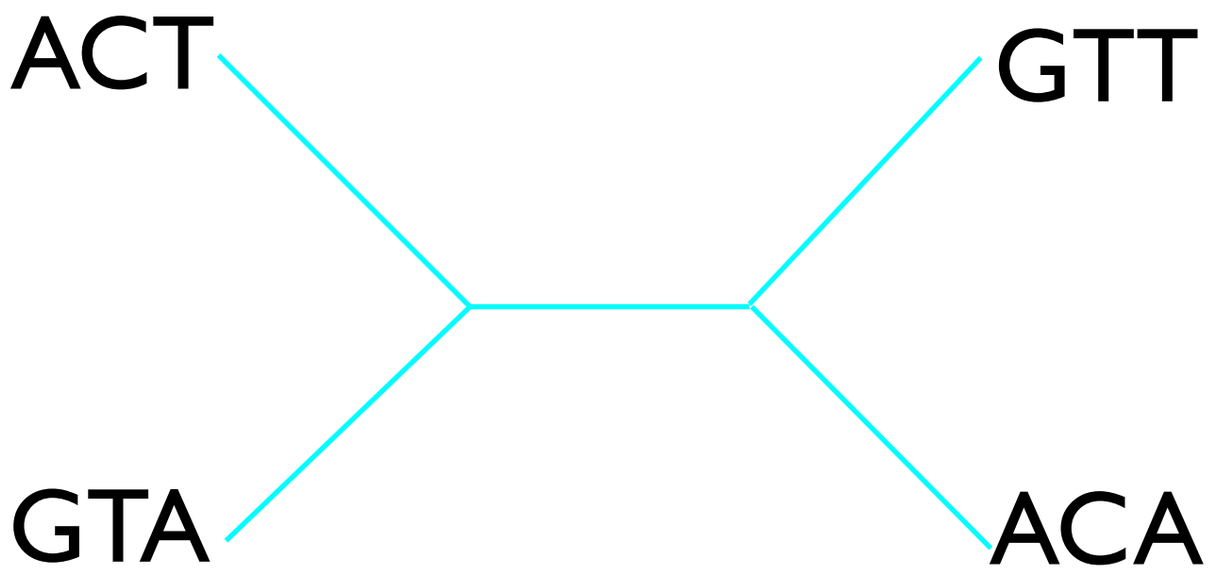
The three trees are equally good MP trees

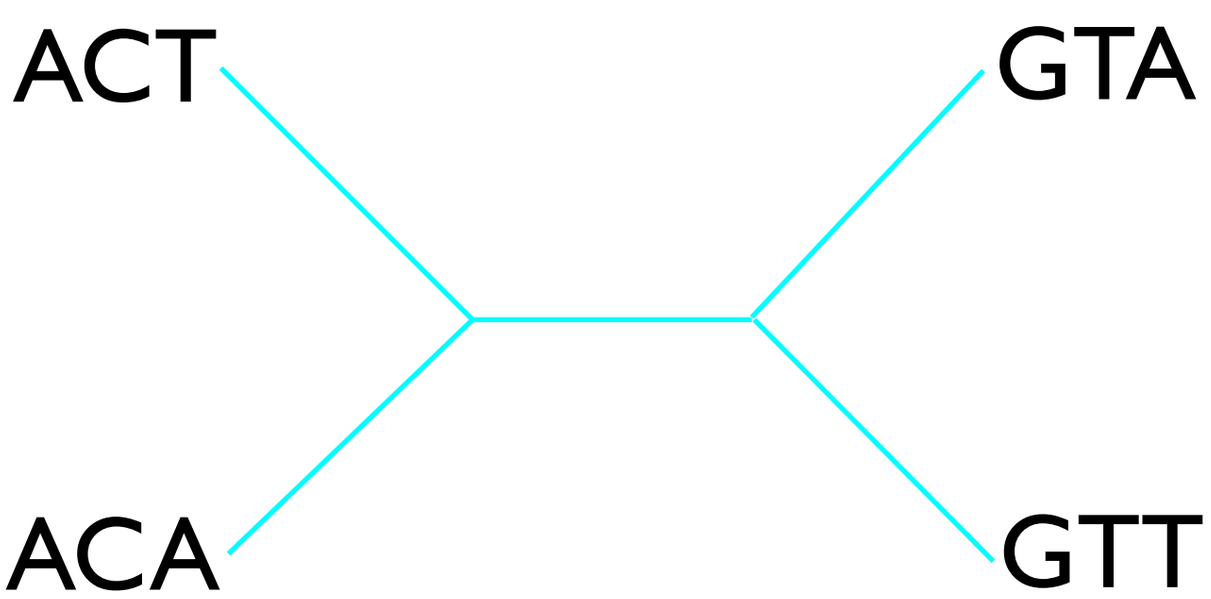


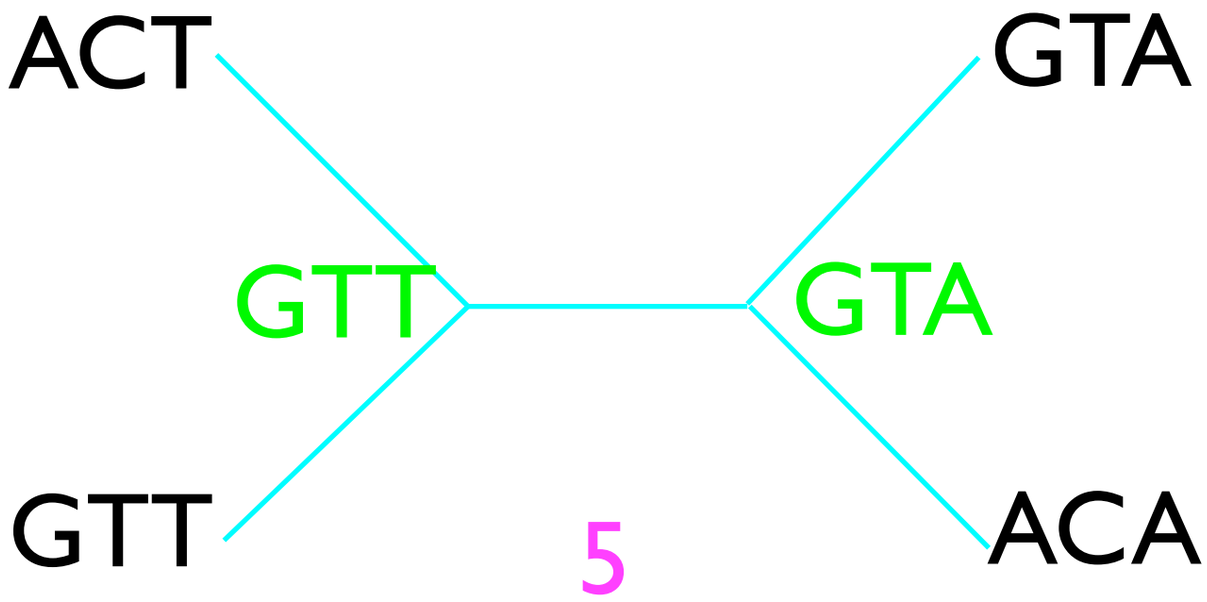


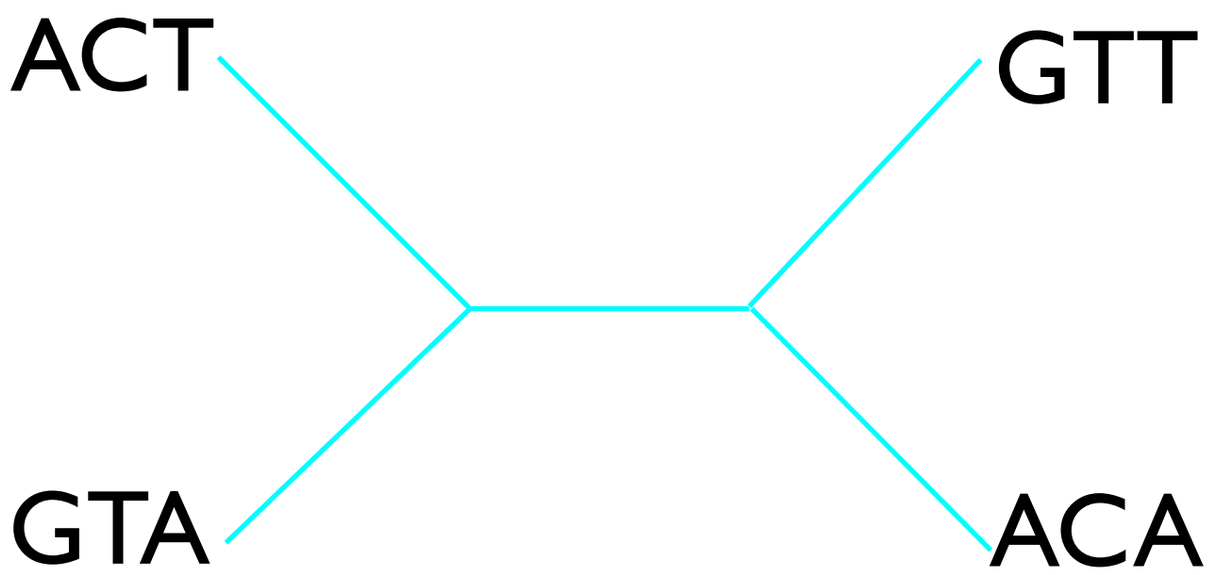
ACT GTT GTA ACA

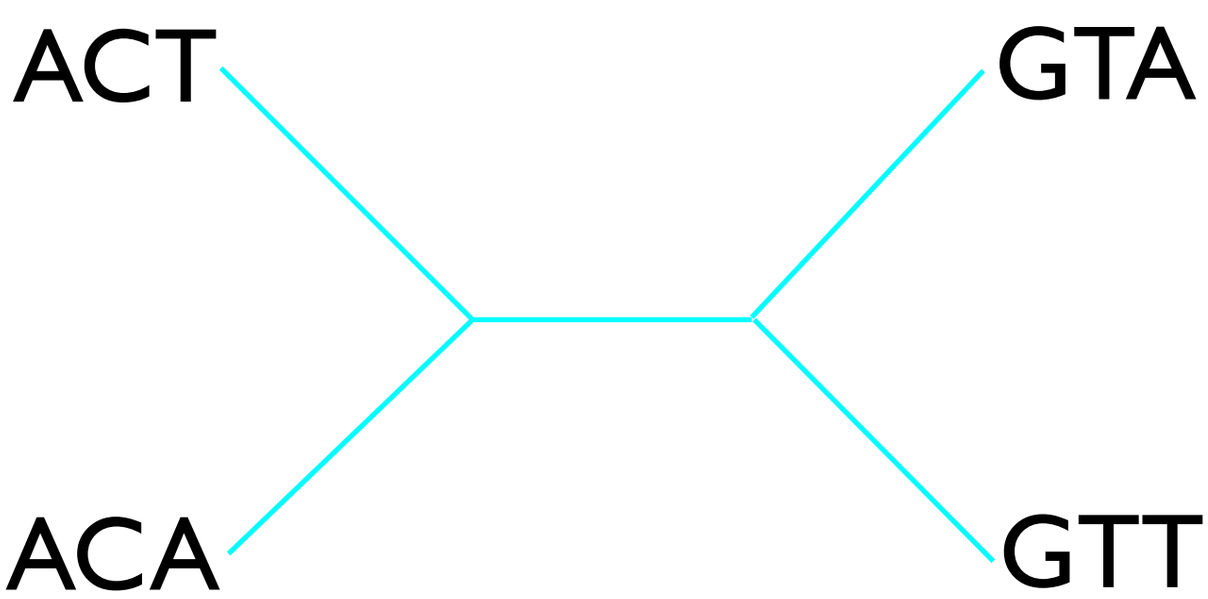

ACT
GTA
GTT
ACA

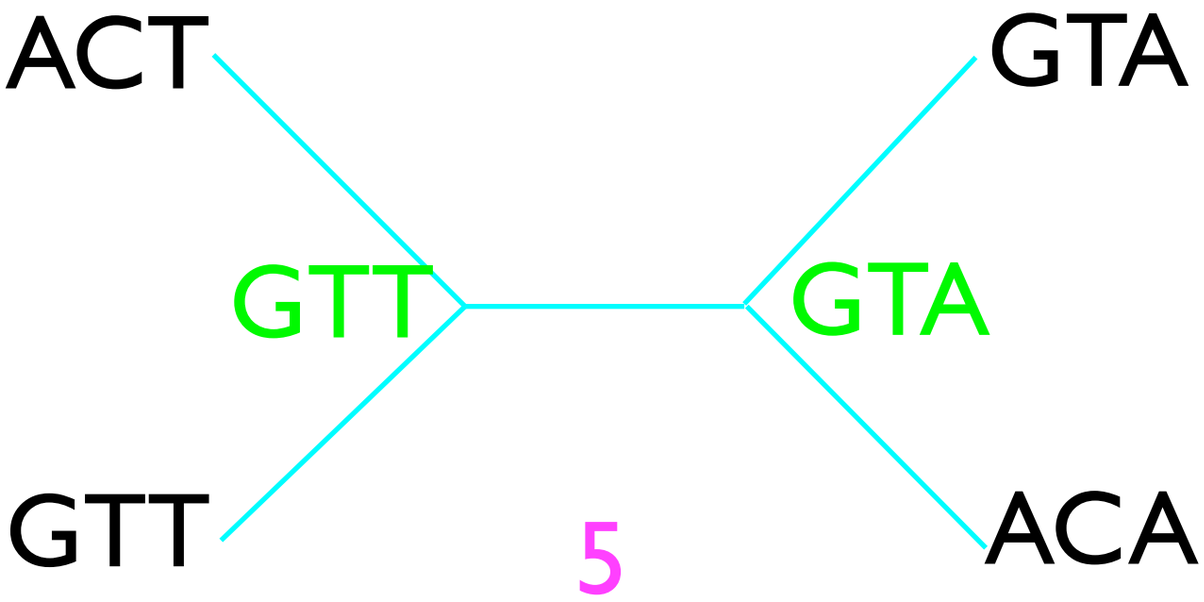

ACT
GTT
GTA
ACA

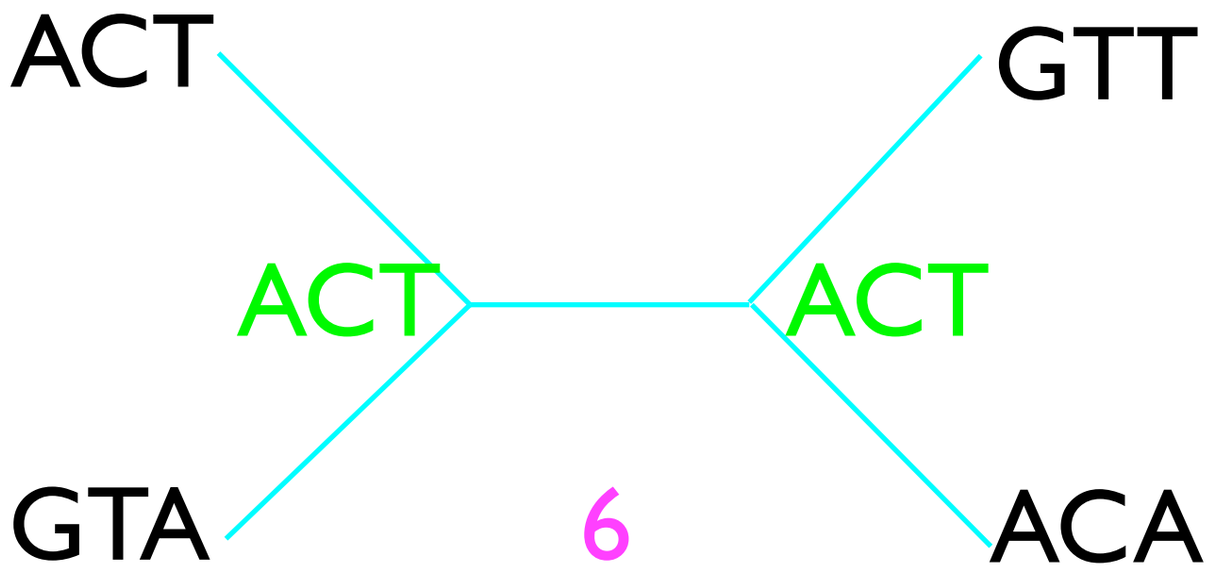

ACT
GTA
ACA
GTT

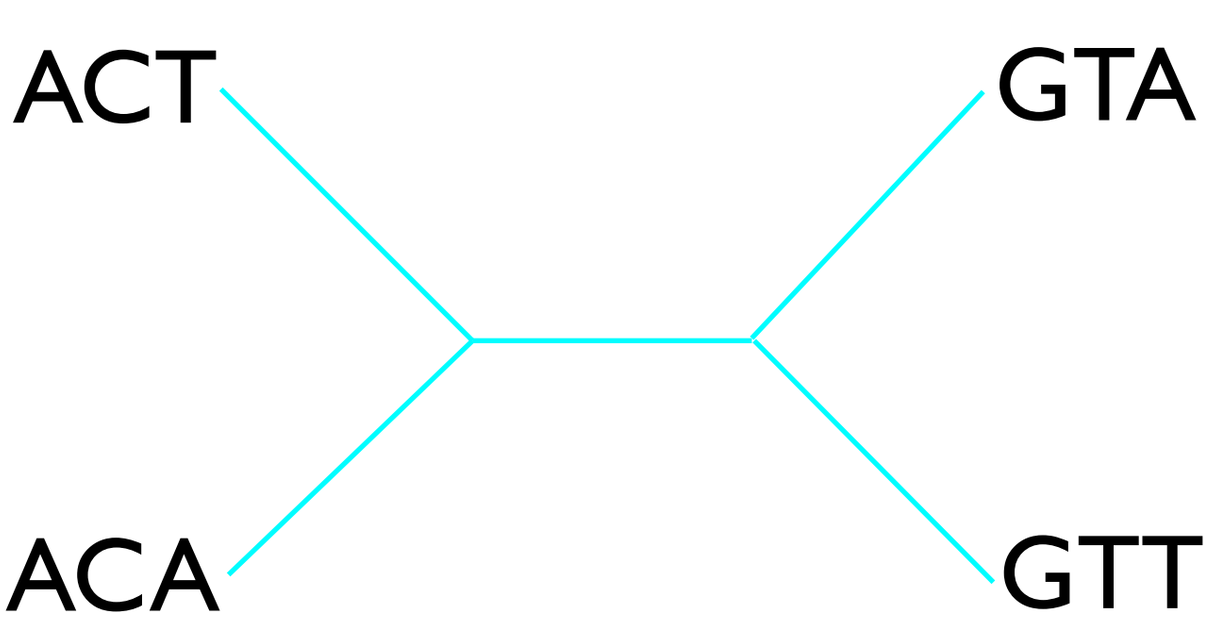

ACT
GTA
GTT
GTA
GTT
ACA
5

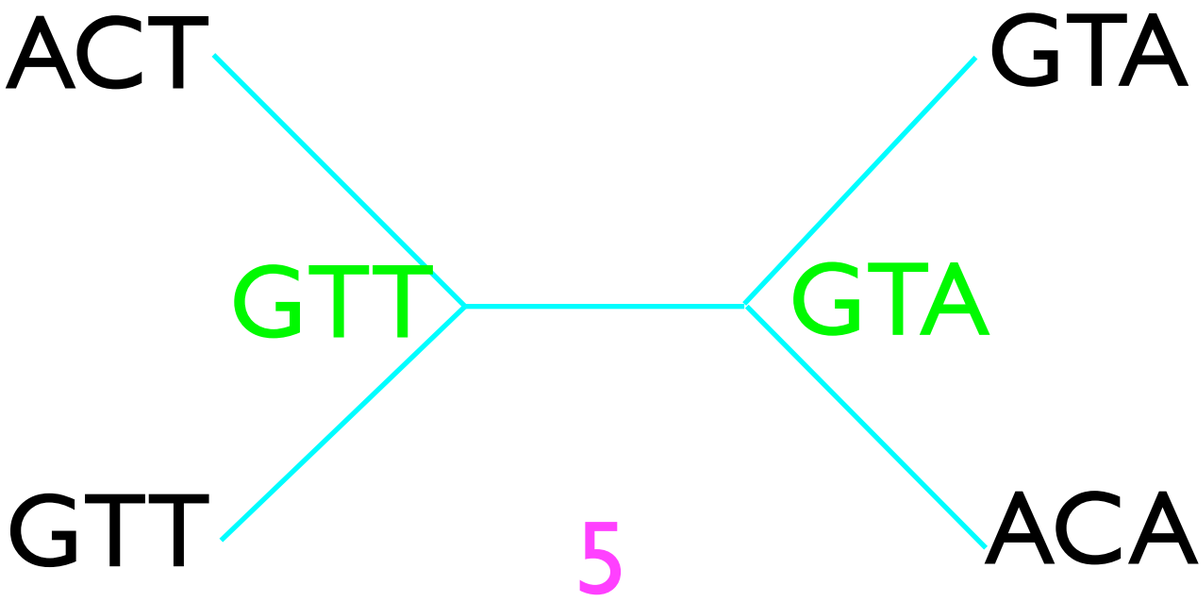

ACT
GTT
GTA
ACA

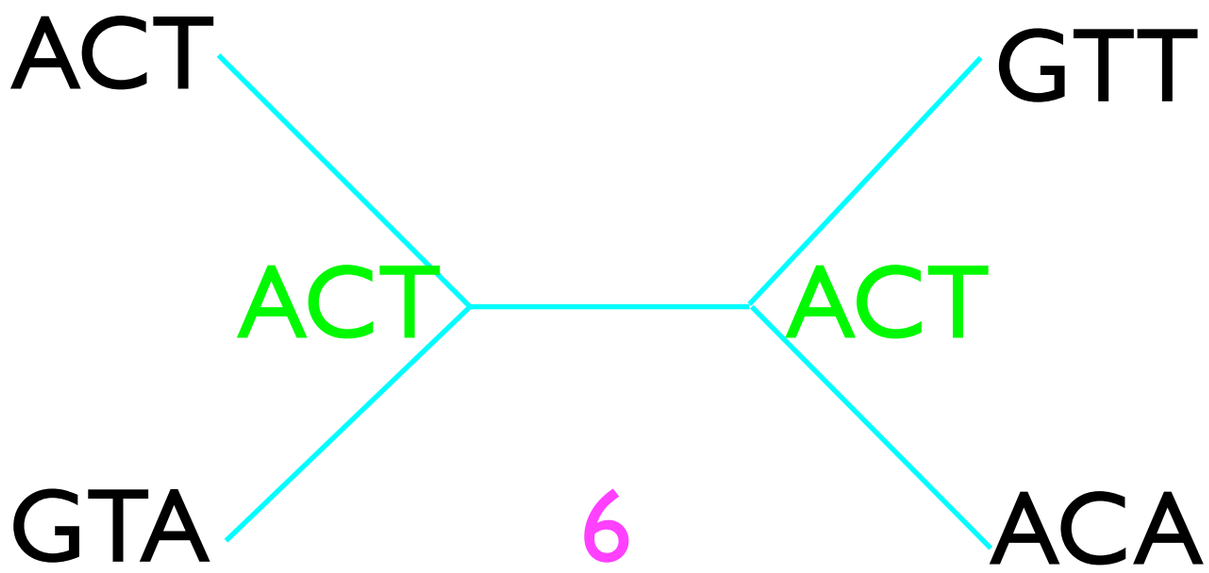

ACT
GTA
ACA
GTT

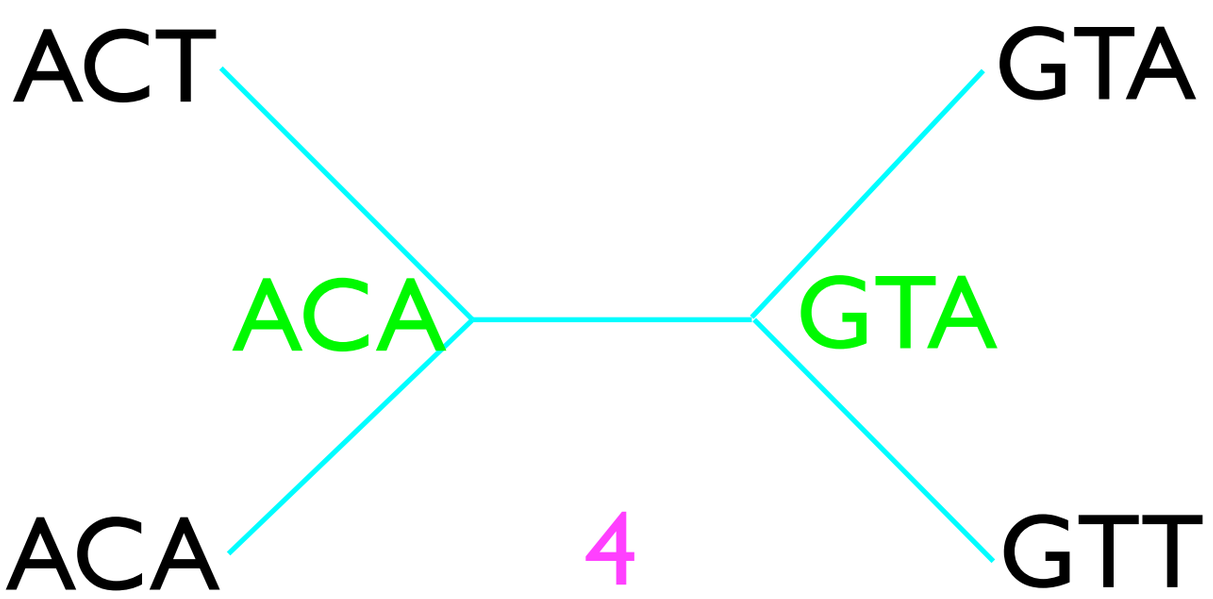

ACT
GTA
GTT
GTA
GTT
ACA
5

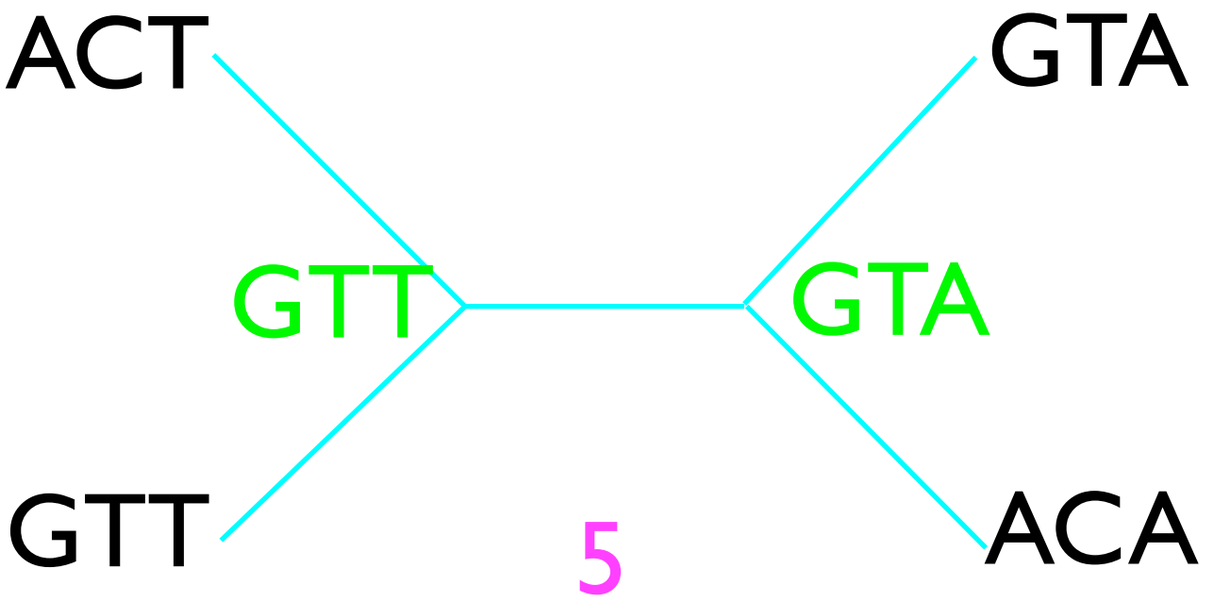

ACT
GTT
ACT
ACT
GTA
6
ACA

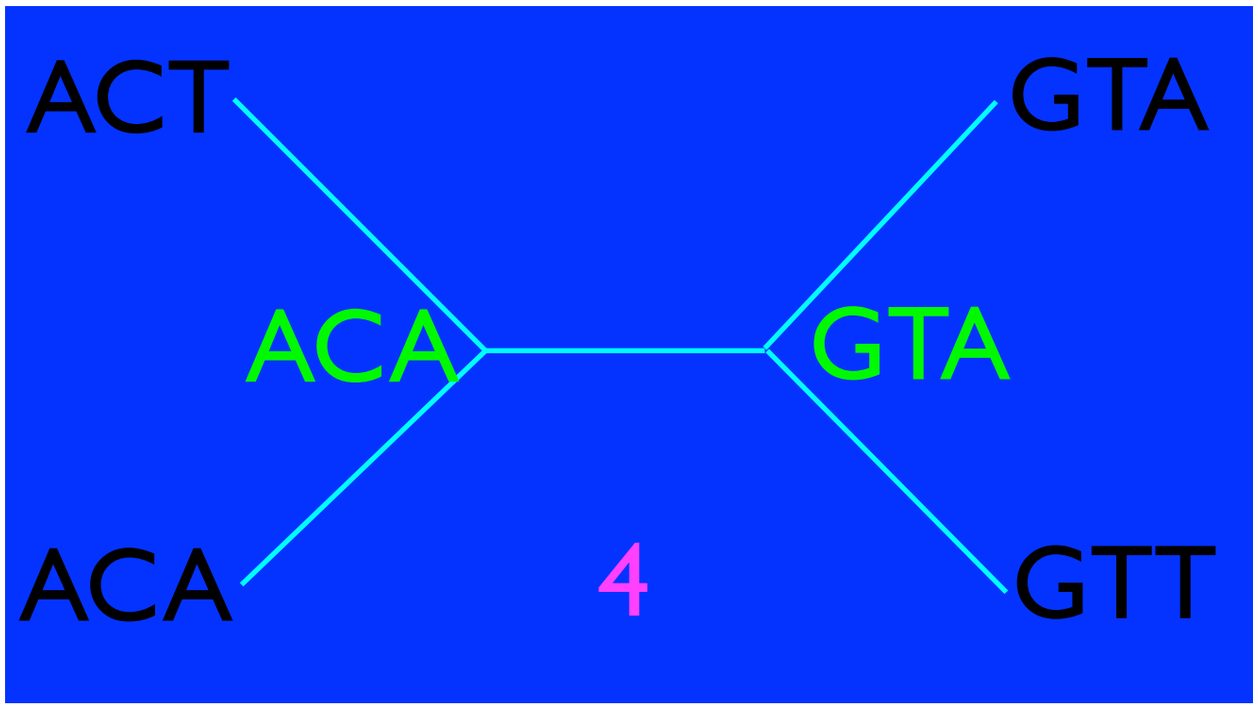

ACT
GTA
ACA
GTT

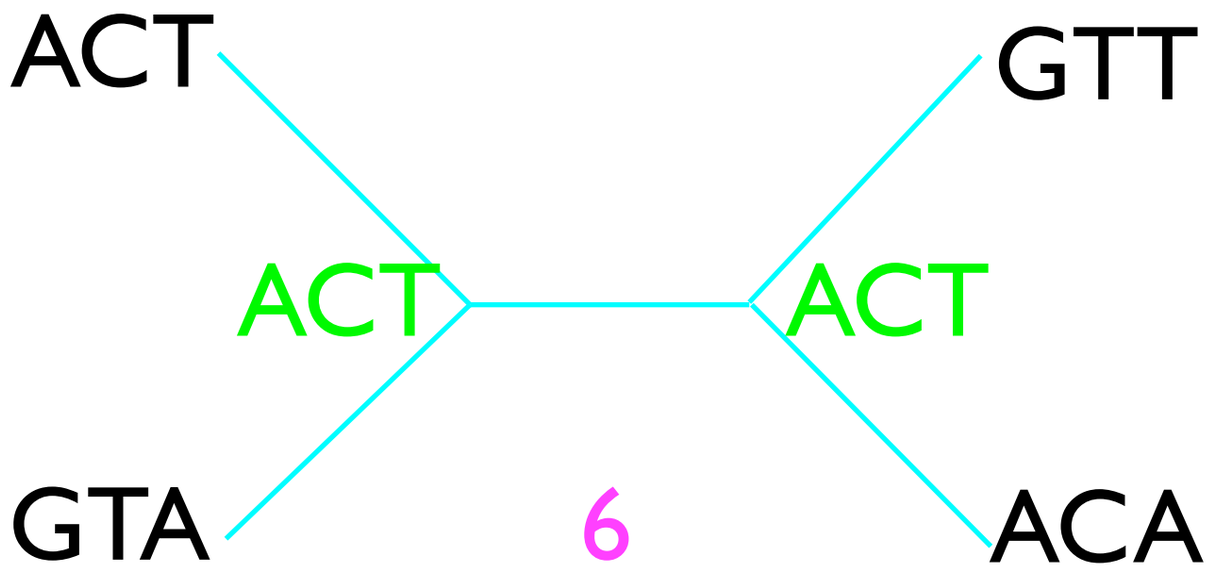

ACT
GTA
GTT
GTA
GTT
ACA
5


ACT
GTT
ACT
ACT
GTA
6
ACA


ACT
GTA
ACA
GTA
ACA
GTT
4


ACT
GTA
GTT
GTA
GTT
ACA
5


ACT
GTA
ACA
GTA
ACA
GTT
4


ACT
GTT
ACT
ACT
GTA
ACA
6

MP tree

Weighted Parsimony

- Each transition from one character state to another is given a weight
- Each character is given a weight
- See a tree that minimizes the weighted parsimony

- Both the MP and weighted MP problems are NP-hard

A Heuristic For Solving the MP Problem

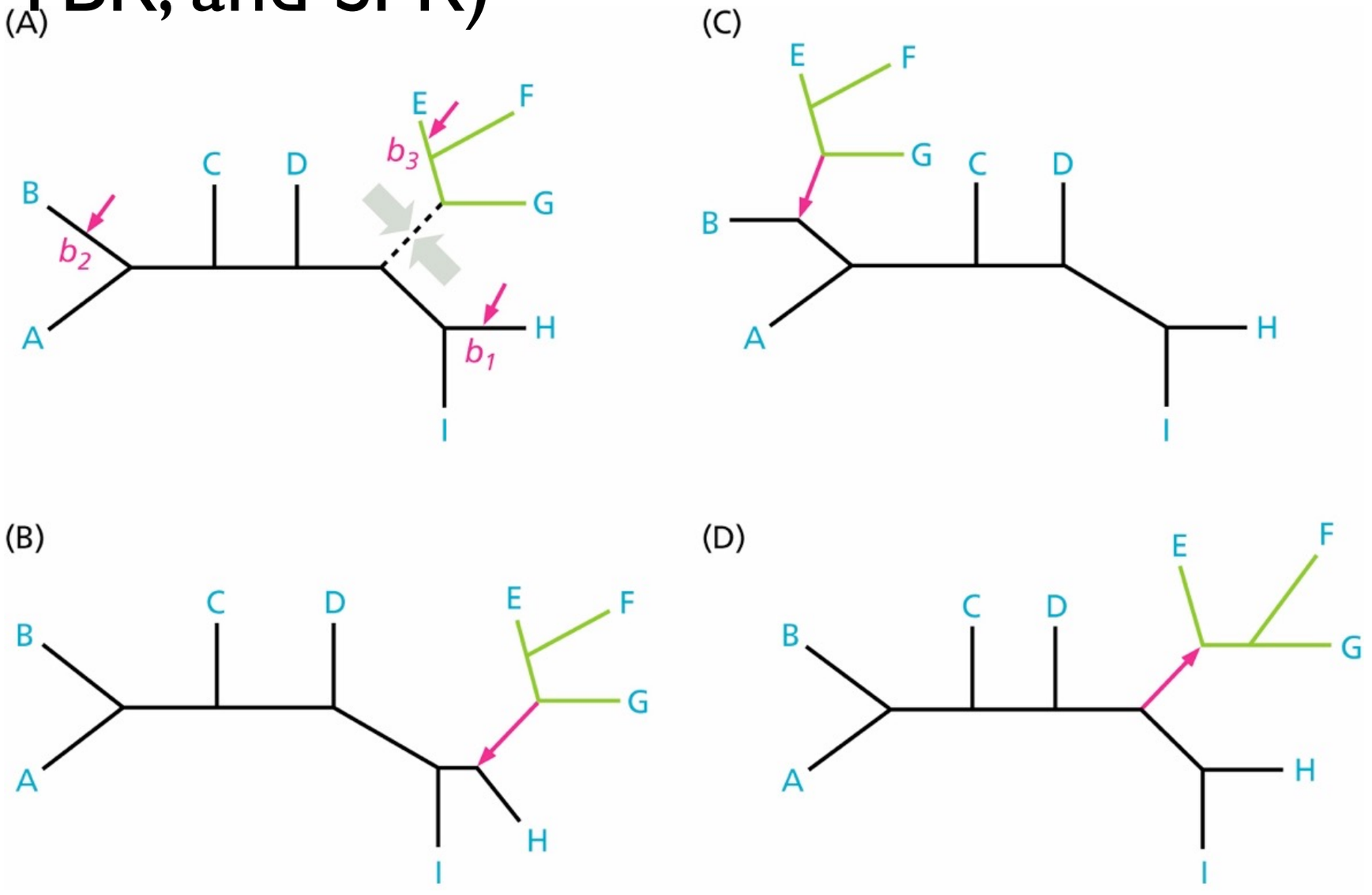
- Starting with a random tree *T*, move through the tree space while computing the parsimony of trees, and keeping those with optimal score (among the ones encountered)
- Usually, the search time is the stopping factor

Two Issues

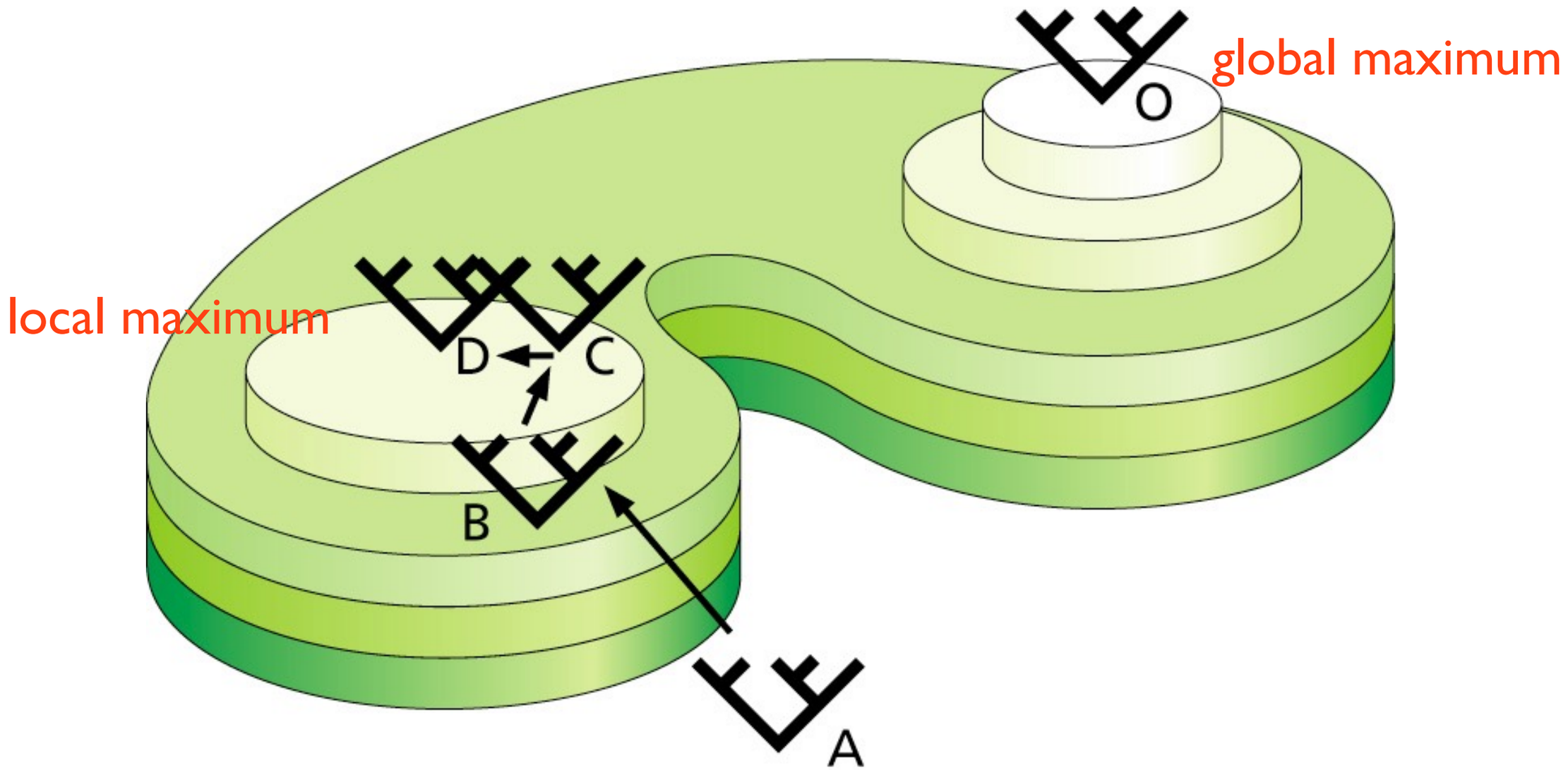
- How do we move through the tree search space?
- Can we compute the parsimony of a given leaf-labeled tree efficiently?

Searching Through the Tree Space

- Use tree transformation operations (NNI, TBR, and SPR)



Searching Through the Tree Space



Computing the Parsimony Length of a Given Tree

- Fitch's algorithm
- Computes the parsimony score of a given leaf-labeled rooted tree
- Polynomial time

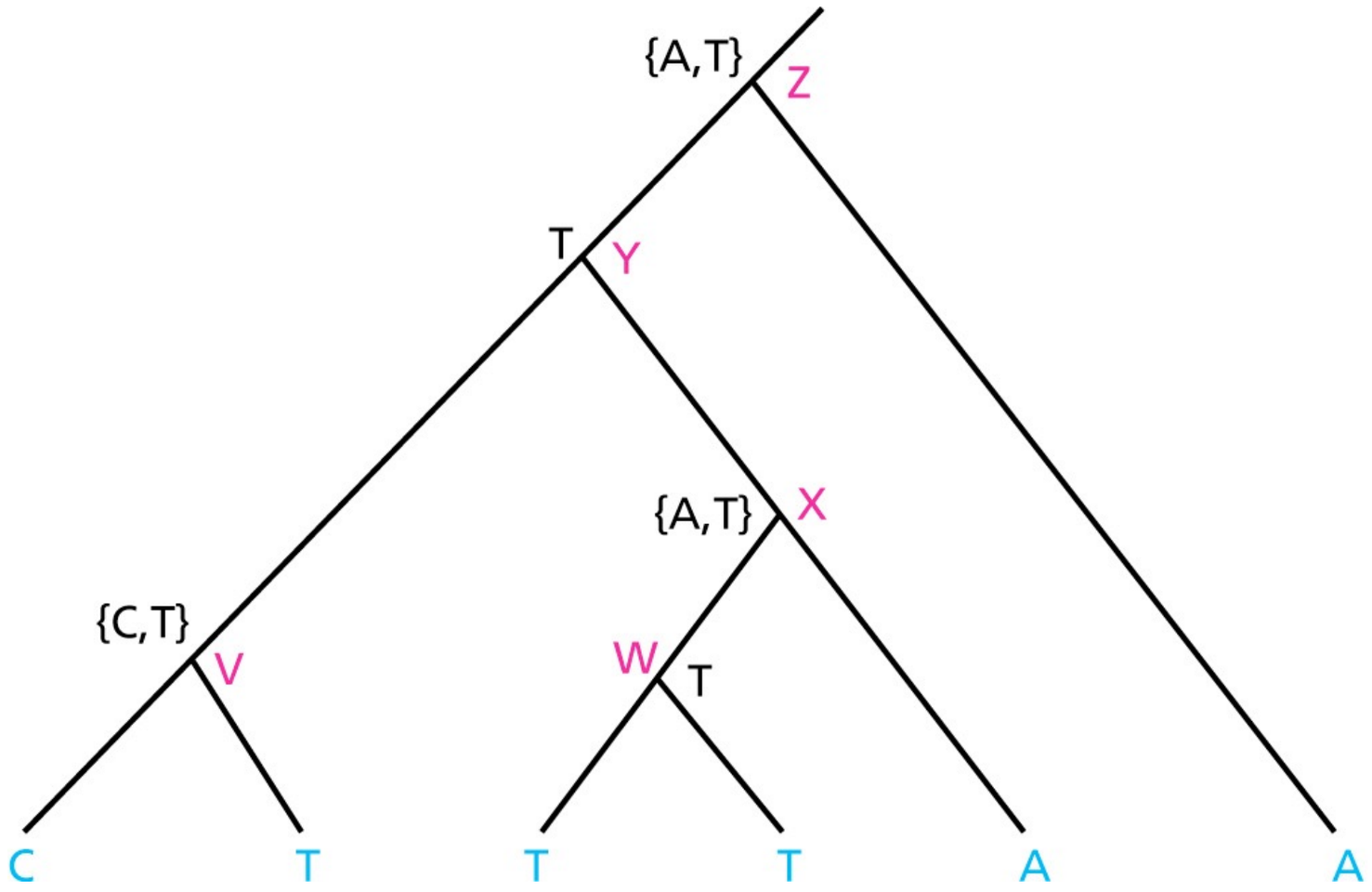
Fitch's Algorithm

- Alphabet Σ
- Character c takes states from Σ
- v_c denotes the state of character c at node v

Fitch's Algorithm

- <u>Bottom-up phase:</u>
- For each node *v* and each character *c*, compute the set $S_{c,v}$ as follows:
  - If *v* is a leaf, then $S_{c,v}=\{v_c\}$
  - If *v* is an internal node whose two children are *x* and *y*, then

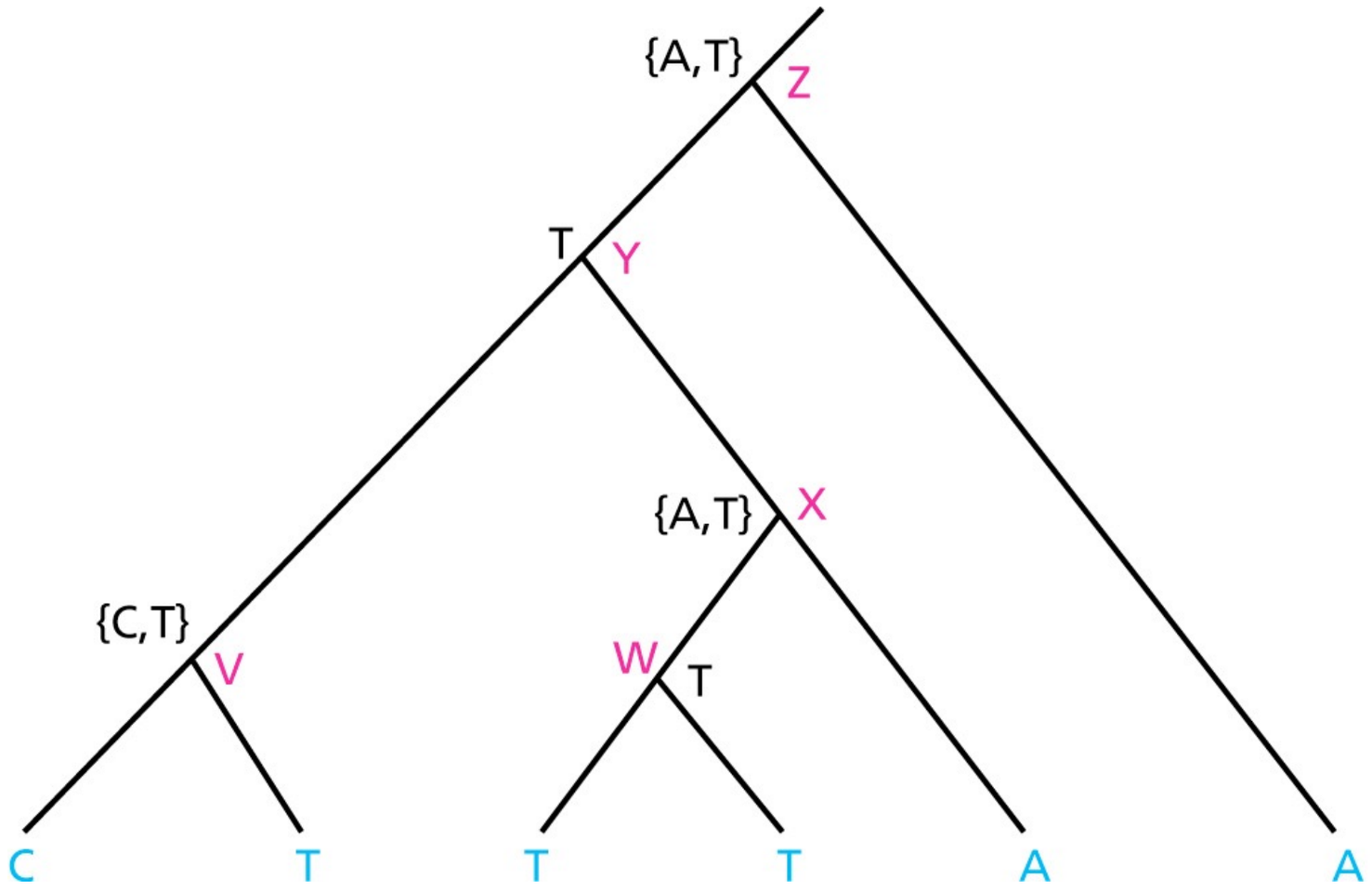
$$S_{c,v} = \begin{cases} S_{c,x} \cap S_{c,y} & S_{c,x} \cap S_{c,y} \neq \emptyset \\ S_{c,x} \cup S_{c,y} & \text{otherwise} \end{cases}$$

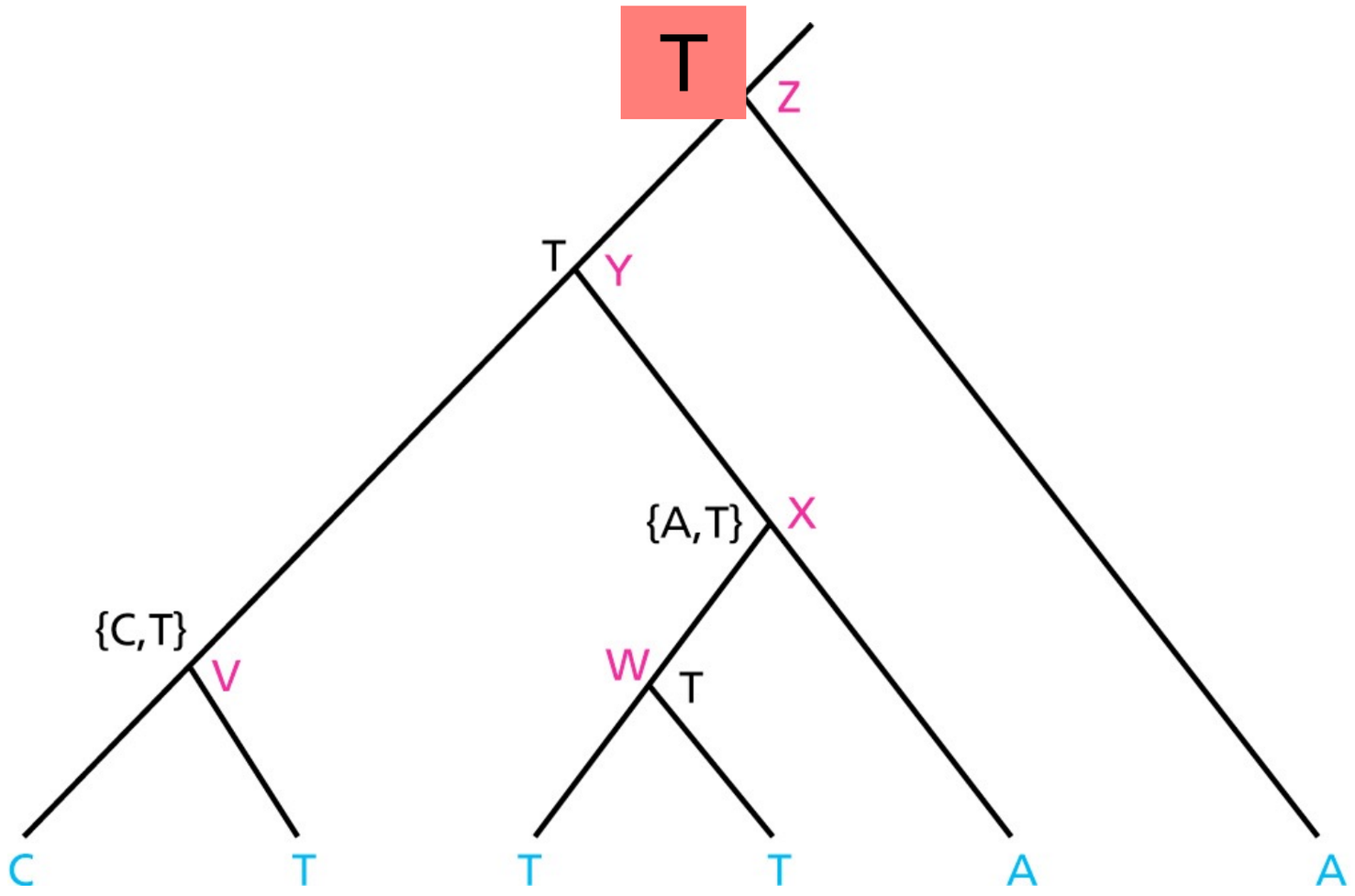

{A,T}
Z
T
Y
{A,T}
X
{C,T}
V
W
T
C
T
T
T
A
A

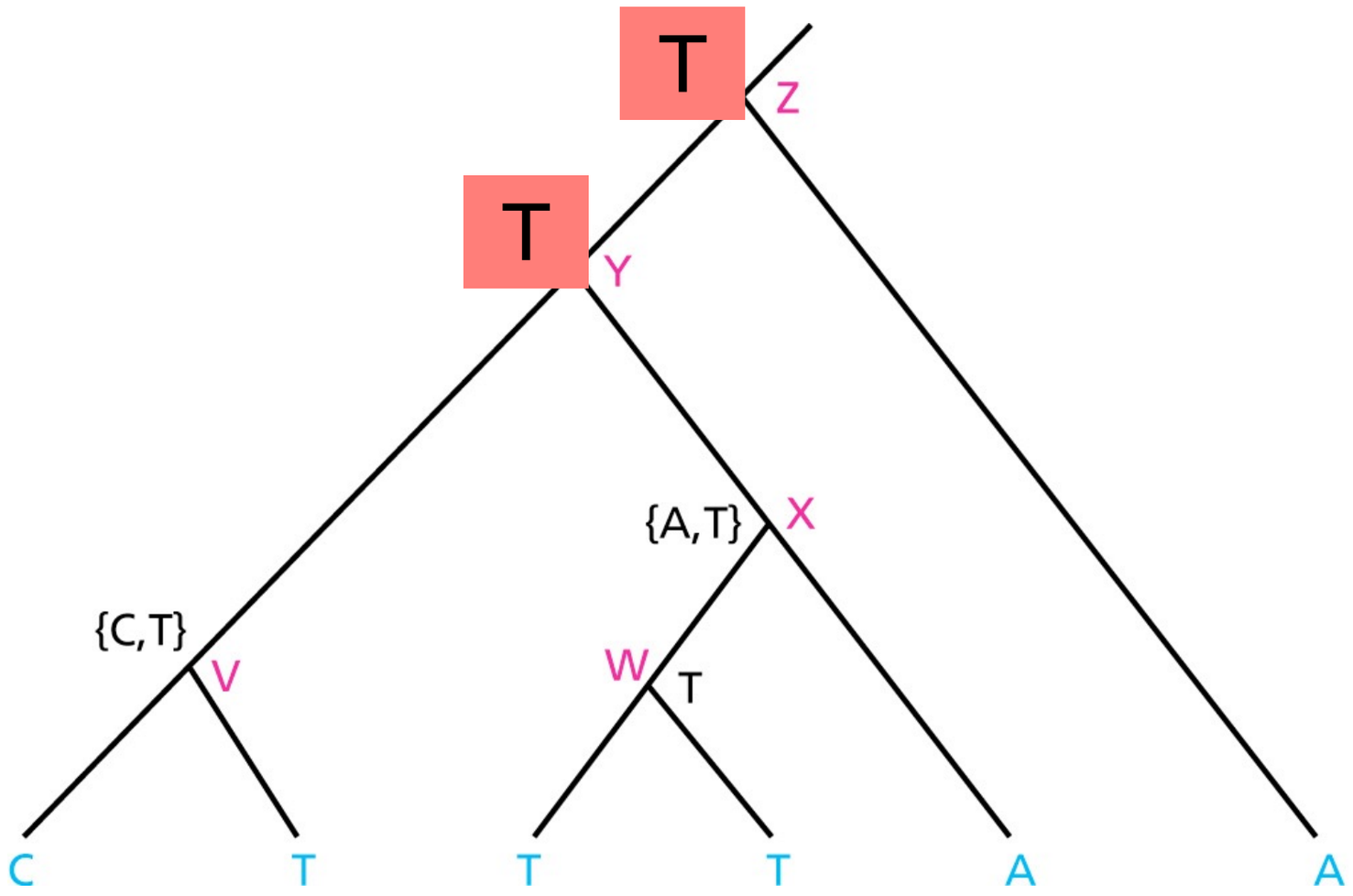
Fitch's Algorithm

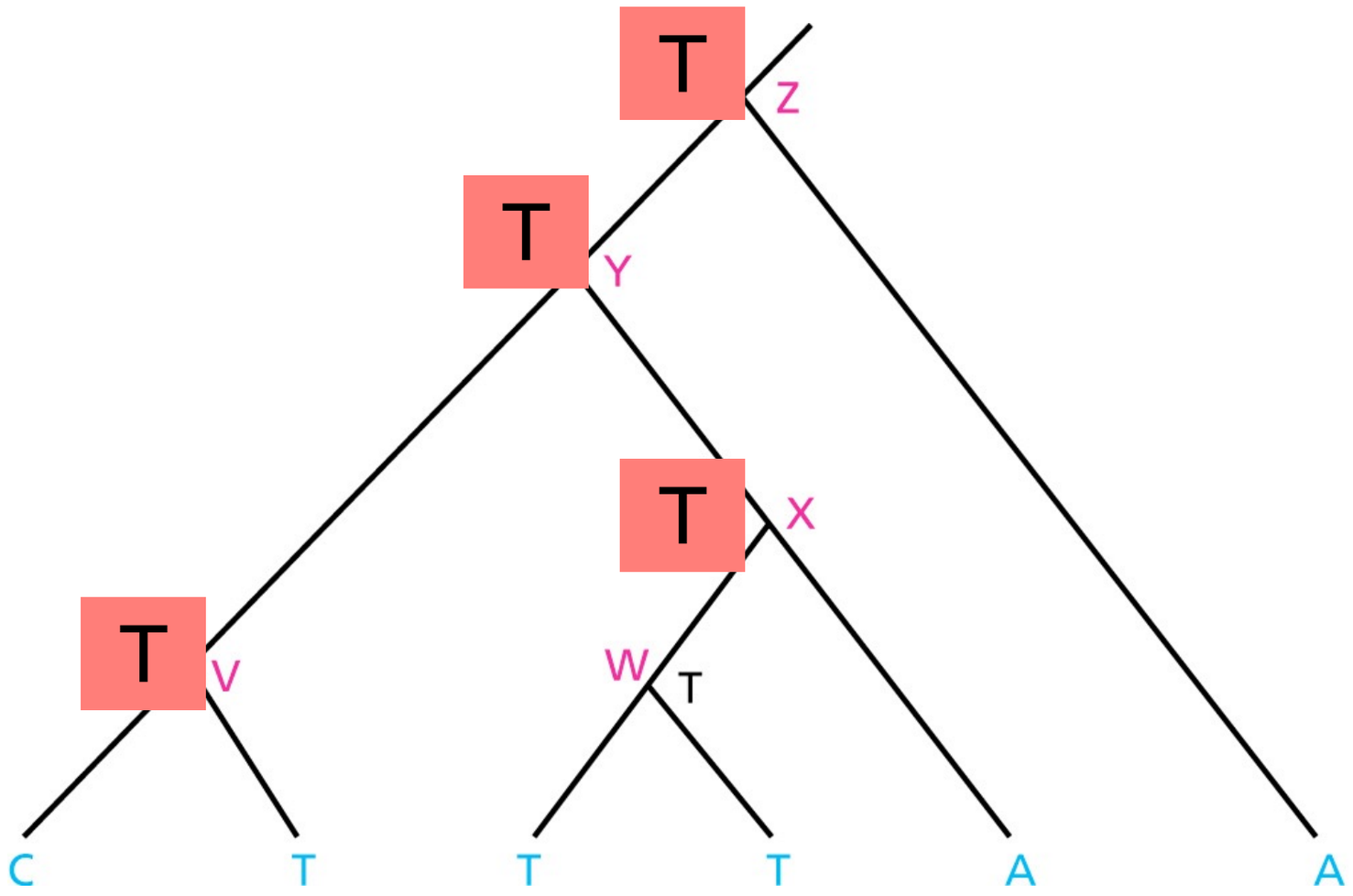
- <u>Top-down phase:</u>
- For the root *r*, let r_c=*a* for some arbitrary *a* in the set $S_{c,r}$
- For internal node *v* whose parent is *u*,

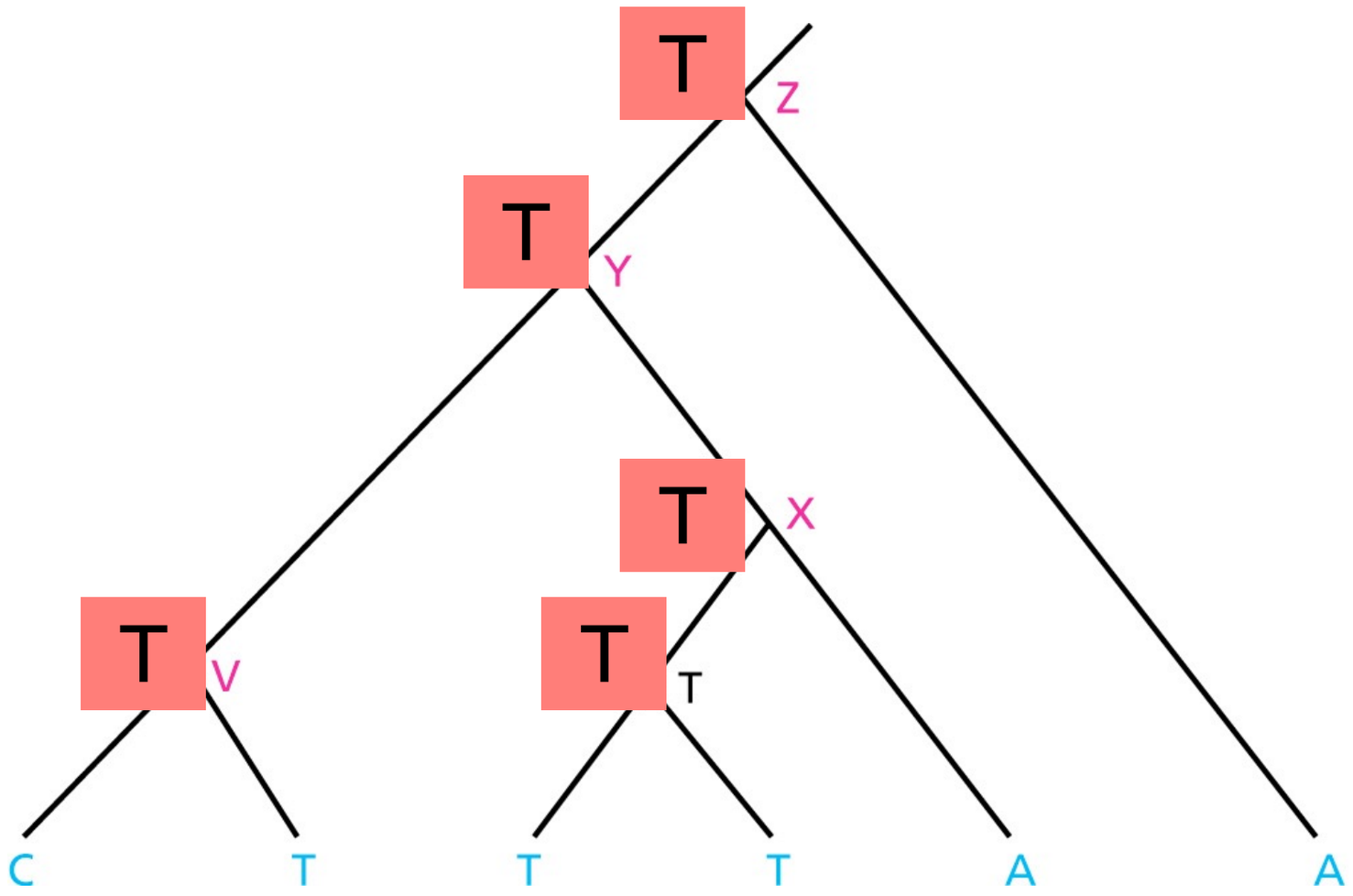
$$v_c = \begin{cases} u_c & u_c \in S_{c,v} \\ \text{arbitrary } \alpha \in S_{c,v} & \text{otherwise} \end{cases}$$

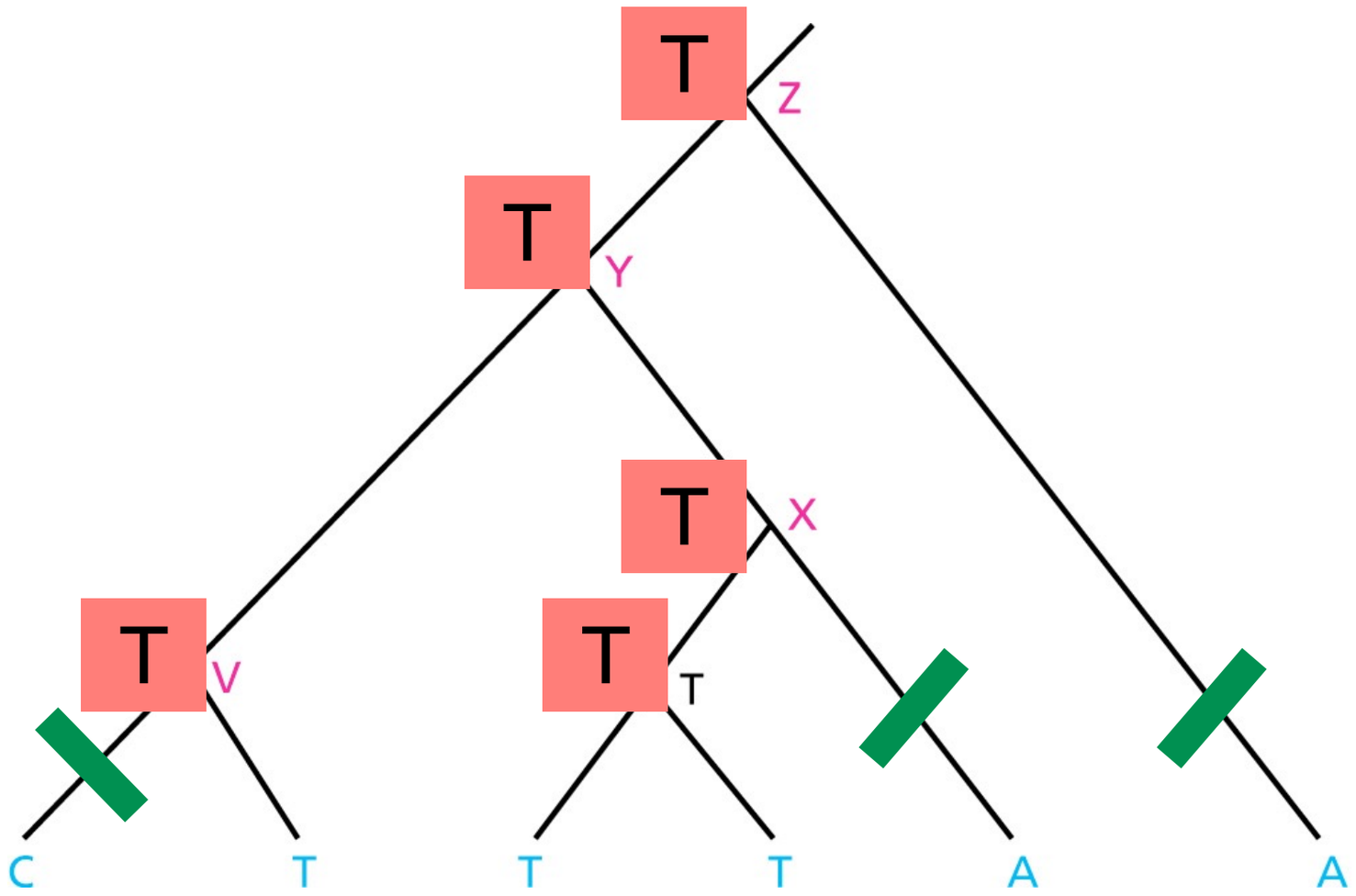

{A,T}
Z
T
Y
{A,T}
X
{C,T}
V
W
T
C
T
T
T
A
A


T
Z
T
Y
{C,T}
V
{A,T}
X
W
T
C
T
T
T
A
A


T
Z
T
Y
{A,T}
X
{C,T}
V
W
T
C
T
T
T
A
A


T
Z
T
Y
T
V
C
T
T
X
W
T
T
T
A
A


T
Z
T
Y
T
V
C
T
T
X
T
T
T
T
A
A



3 mutations

Fitch's Algorithm

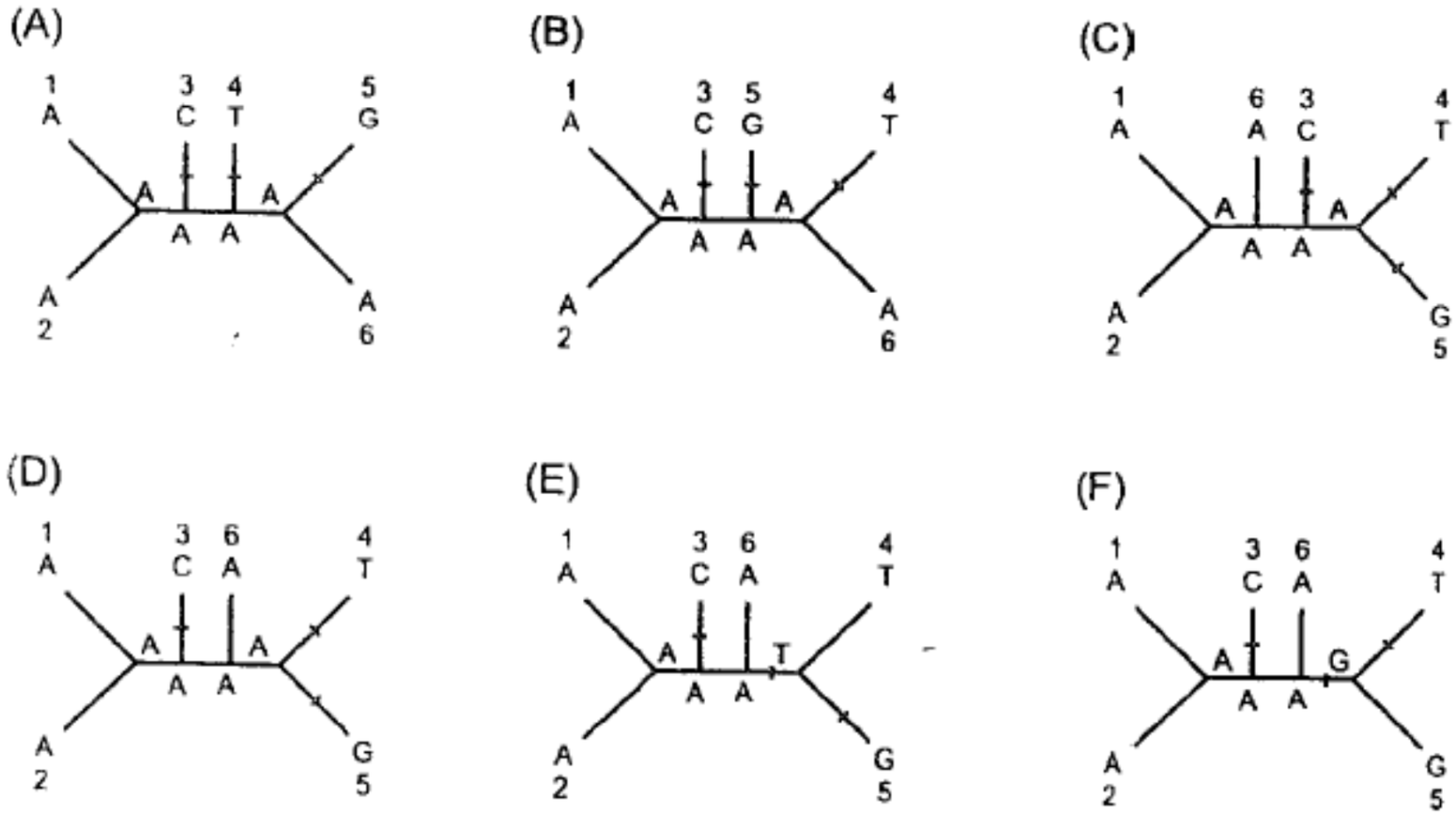
- Takes time O(*nkm*), where *n* is the number of leaves in the tree, *m* is the number of sites, and *k* is the maximum number of states per site (for DNA, *k*=4)

Informative Sites and Homoplasy

- Invariable sites: In the search for MP trees, sites that exhibit exactly one state for all taxa are eliminated from the analysis
- Only variable sites are used

Informative Sites and Homoplasy

- However, not all variable sites are useful for finding an MP tree topology
- Singleton sites: any nucleotide site at which only unique nucleotides (singletons) exist is not informative, because the nucleotide variation at the site can always be explained by the same number of substitutions in all topologies



C,T,G are three singleton substitutions ⇒non-informative site

All trees have parsimony score 3

Informative Sites and Homoplasy

- For a site to be informative for constructing an MP tree, it must exhibit at least two different states, each represented in at least two taxa
- These sites are called <u>informative sites</u>
- For constructing MP trees, it is sufficient to consider only informative sites

Informative Sites and Homoplasy

- Because only informative sites contribute to finding MP trees, it is important to have many informative sites to obtain reliable MP trees
- However, when the extent of homoplasy (backward and parallel substitutions) is high, MP trees would not be reliable even if there are many informative sites available

Measuring the Extent of Homoplasy

- The consistency index (Kluge and Farris, 1969) for a single nucleotide site (*i*-th site) is given by $c_i = m_i / s_i$, where

  - m_i is the minimum possible number of substitutions at the site for any conceivable topology (= one fewer than the number of different kinds of nucleotides at that site, assuming that one of the observed nucleotides is ancestral)

  - s_i is the minimum number of substitutions required for the topology under consideration

Measuring the Extent of Homoplasy

- The lower bound of the consistency index is not 0
- The consistency index varies with the topology
- Therefore, Farris (1989) proposed two more quantities: the retention index and the rescaled consistency index

The Retention Index

- The retention index, r_i, is given by $(g_i-s_i)/(g_i-m_i)$,

  where g_i is the maximum possible number of substitutions at the i-th site for any conceivable tree under the parsimony criterion and is equal to the number of substitutions required for a star topology when the most frequent nucleotide is placed at the central node

The Retention Index

- The retention index becomes 0 when the site is least informative for MP tree construction, that is, $s_i = g_i$

The Rescaled Consistency Index

$$rc_i = \frac{g_i - s_i \; m_i}{g_i - m_i \; s_i}$$

Ensemble Indices

- The three values are often computed for all informative sites, and the ensemble or overall consistency index (CI), overall retention index (RI), and overall rescaled index (RC) for all sites are considered

Ensemble Indices

$$CI = \frac{\sum_i m_i}{\sum_i s_i}$$

$$RI = \frac{\sum_i g_i - \sum_i s_i}{\sum_i g_i - \sum_i m_i}$$

$$RC = CI \times RI$$

These indices should be computed only for informative sites, because for uninformative sites they are undefined

Homoplasy Index

- The homoplasy index is $HI = 1 - CI$
- When there are no backward or parallel substitutions, we have $HI = 0$. In this case, the topology is uniquely determined

Warning!

- Maximum parsimony is not statistically consistent!

Likelihood

- The likelihood of model M given data D, denoted by L(M|D), is p(D|M).
- For example, consider the following data D that result from tossing a coin 10 times:
  - HTTTTHTTTT

- Model M1:
  - A fair coin (p(H)=p(T)=0.5)
  - $L(M1|D)=p(D|M1)=0.5^{10}$

- Model M2:
  - A biased coin (p(H)=0.8,p(T)=0.2)
  - $L(M2|D)=p(D|M2)=0.8^2 0.2^8$

- Model M3:
  - A biased coin (p(H)=0.1,p(T)=0.9)
  - $L(M3|D)=p(D|M3)=0.1^2 0.9^8$

- The problem of interest is to infer the model M from the (observed) data D.

- The maximum likelihood estimate, or MLE, is:

$$\hat{M} \leftarrow \text{argmax}_M p(D|M)$$

- D=HTTTTTHTTTT
- M1: p(H)=p(T)=0.5
- M2: p(H)=0.8, p(T)=0.2
- M3: p(H)=0.1, p(T)=0.9
- MLE (among the three models) is M3.

- A more complex example:
  - The model M is an HMM
  - The data D is a sequence of observations
  - Baum-Welch is an algorithm for obtaining the MLE M from the data D

- The model parameters that we seek to learn can vary for the same data and model.

- For example, in the case of HMMs:

    - The parameters are the states, the transition and emission probabilities (no parameter values in the model are known)

    - The parameters are the transition and emission probabilities (the states are known)

    - The parameters are the transition probabilities (the states and emission probabilities are known)

Back to Phylogenetic Trees

- What are the data D?
- A multiple sequence alignment
- (or, a matrix of taxa/characters)

Back to Phylogenetic Trees

- What is the (generative) model M?
  - The tree topology
  - The branch lengths
  - The model of evolution (JC, ..)

Back to Phylogenetic Trees

- What is the (generative) model M?
  - The tree topology, T
  - The branch lengths, λ
  - The model of evolution (JC, ..), E

Back to Phylogenetic Trees

- The likelihood is p(D|T,λ,E).
- The MLE is

$$(\hat{T}, \hat{\lambda}, \hat{E}) \leftarrow \text{argmax}_{(T,\lambda,E)} p(D|T, \lambda, E)$$

Back to Phylogenetic Trees

- In practice, the model of evolution is estimated from the data first, and in the phylogenetic inference it is assumed to be known.
- In this case, given D and E, the MLE is

$$(\hat{T}, \hat{\lambda}) \leftarrow \text{argmax}_{(T,\lambda)} p(D|T, \lambda)$$

Assumptions

- Characters are independent
- Markov process: probability of a node having a given label depends only on the label of the parent node and branch length between them t

Maximum Likelihood

- Input: a matrix D of taxa-characters
- Output: tree T leaf-labeled by the set of taxa, and with branch lengths λ so as to maximize the likelihood $P(D|T,\lambda)$

P(D|T,λ)

$$
\begin{aligned}
P(D|T,\lambda) &= \prod_{site\ j} p(D_j|T,\lambda) \\
&= \prod_{site\ j} \left(\sum_R p(D_j,R|T,\lambda)\right) \\
&= \prod_{site\ j} \left(\sum_R \left[p(root)\cdot \prod_{edge\ u\to v} p_{u\to v}(t_{uv})\right]\right)
\end{aligned}
$$

- What is $p_{ij}(t_{uv})$ for a branch u→v in the tree, where i and j are the states of the site at nodes u and v, repsectively?

- Assume that in a small interval of time of length dt, there is a probability (α dt) that the current base at a site is replaced.

- The quantity α is the rate of base substitution per unit time.

- If a base is replaced, its replacement is A, C, G, or T with probabilities π_1, π_2, π_3, or π_4.

- If we let δ_{ij}=0 if the states at nodes u and v are different, and δ_{ij}=1 if the states at nodes u and v are the same, then

$$p_{ij}(dt) = (1 - \alpha\ dt)\delta_{ij} + \alpha\ dt\ \pi_j$$

- For an arbitrary t_{uv}:

$$p_{ij}(t_{uv}) = e^{-\alpha t_{uv}} \delta_{ij} + (1 - e^{-\alpha t_{uv}}) \pi_j$$

- The ML problem is NP-hard (that is, finding the MLE (T,λ) is very hard computationally)

- Heuristics involve searching the tree space, while computing the likelihood of trees

- Computing the likelihood of a leaf-labeled tree T with branch lengths can be done efficiently using dynamic programming

P(D|T,λ)

Let $C_j(x,v)$ = P(subtree whose root is v | v_j=x)

Initialization: leaf v and state x $\quad C_j(x,v) = \begin{cases} 1 & v_j = x \\ 0 & \text{otherwise} \end{cases}$

Recursion: node v with children u,w

$$C_j(x,v) = \left[\sum_y C_j(y,u) \cdot P_{x\to y}(t_{vu})\right] \cdot \left[\sum_y C_j(y,w) \cdot P_{x\to y}(t_{vw})\right]$$

Termination:

$$L = \prod_{j=1}^{m} \left[\sum_x C_j(x,root) \cdot \mathrm{P}(x)\right]$$

Running Time

- Takes time $O(nk^2m)$, where n is the number of leaves in the tree, m is the number of sites, and k is the maximum number of states per site (for DNA, k=4)