Introduction to Bioinformatics for Computer Scientists

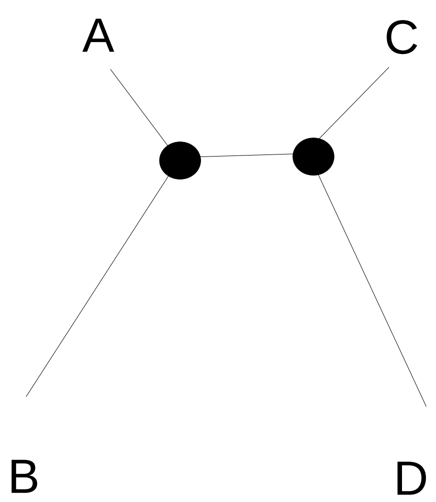
Lecture 10

Outline – Lecture 10

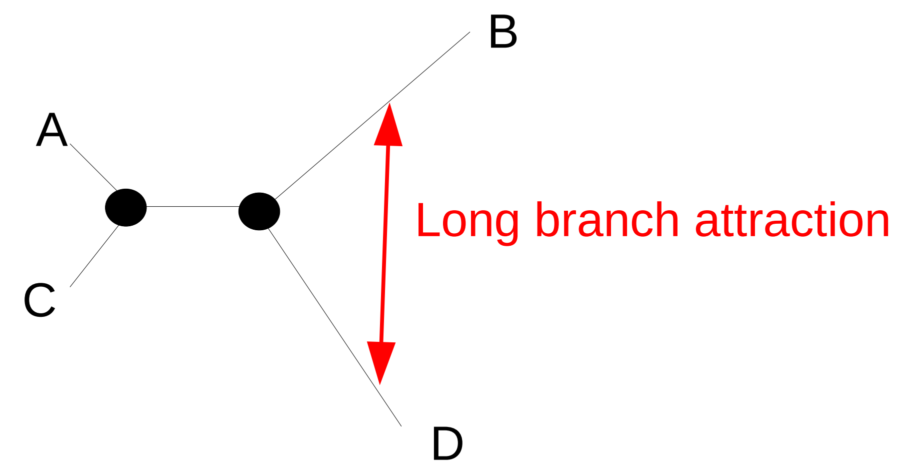
- **Maximum Likelihood – motivation**
- Computing the Likelihood on a tree
- Computing the Maximum Likelihood on a tree

Parsimony & Long Branch Attraction

- Because parsimony tries to minimize the number of mutations it faces some problems on trees with long branches



Correct tree

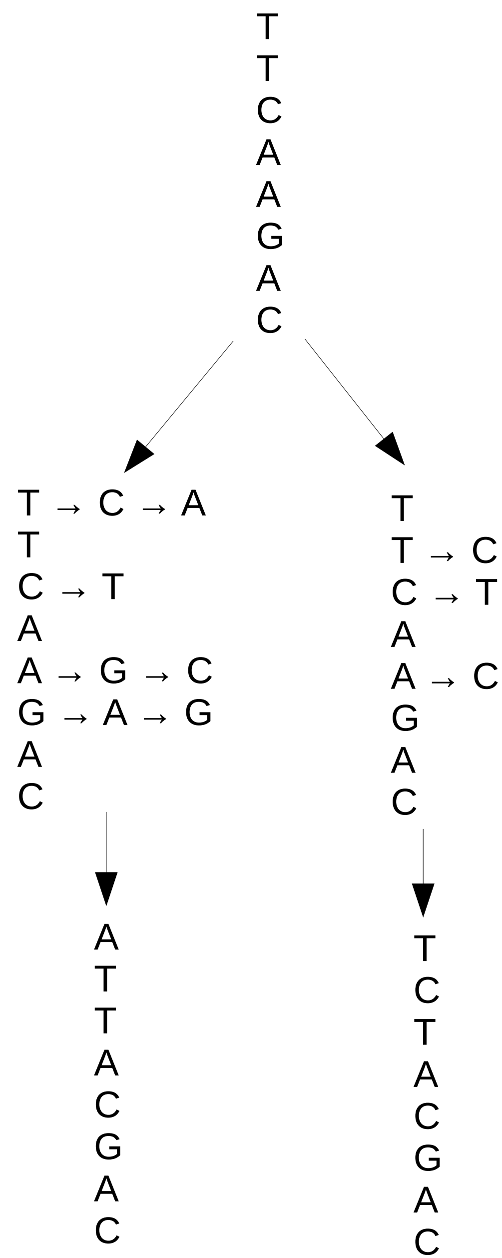


Wrong tree inferred by parsimony

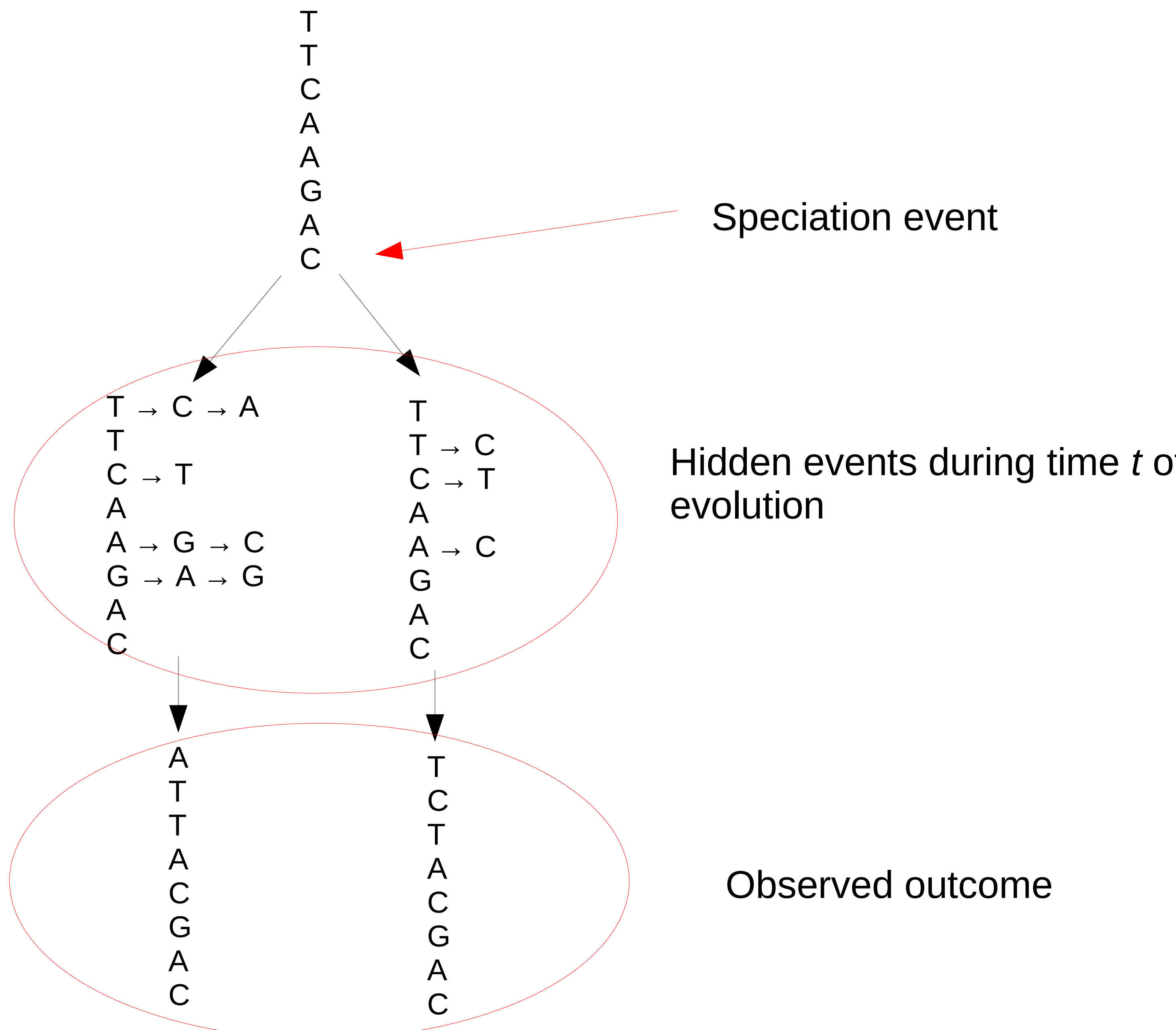
Parsimony & Long Branch Attraction

- Settings under which parsimony recovers the wrong tree are also called **“the Felsenstein Zone”** after *Joe Felsenstein* who has made numerous very important contributions to the field, e.g.
  - The Maximum Likelihood model
  - The Bootstrapping procedure
- If you are interested in statistics, there are some on-line courses by Joe at http://evolution.gs.washington.edu/courses.html

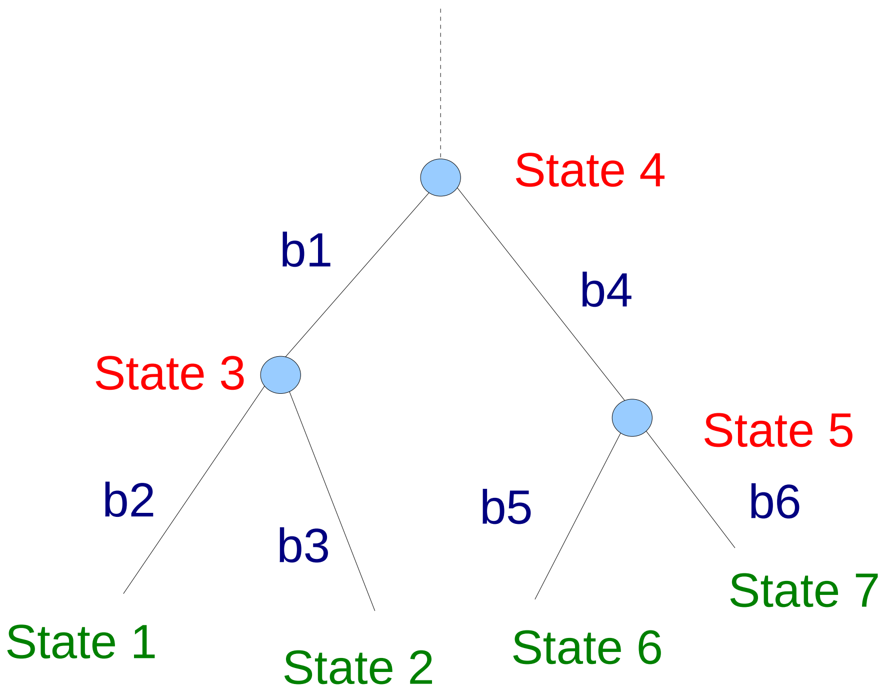
Likelihood tries to fix this Problem



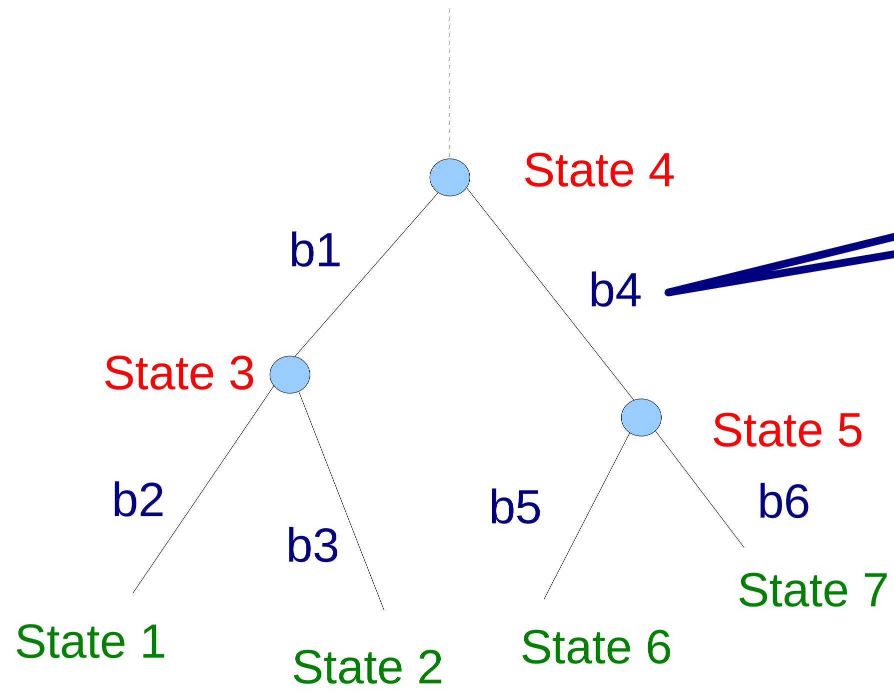
Likelihood tries to fix this Problem



2nd part of lecture → How do we compute the likelihood on a tree?



3rd part of Lecture → How do we maximize the likelihood on a tree?



We need, for instance, to optimize/estimate the branch lengths

Outline – Lecture 10

- Maximum Likelihood – motivation
- **Computing the Likelihood on a tree**
- Computing the Maximum Likelihood on a tree

Likelihood

- Given:
  - MSA
  - Tree topology with branch lengths
  - Model
  - We can calculate $P_{x\text{->}z}(b)$ for a branch length (or time) *b*
  - $P_{x\text{->}z}(b)$ is our continuous time Markov Model of sequence evolution!
  - We obtain $P_{x\text{->}z}(b)$ by exponentiating the instantaneous rate matrix Q

Likelihood

- $L(T|D) = P(D|T)$

Probability that the tree generated the data (generating process)

Likelihood

- $L(T|D) = P(D|T)$

Likelihood of the tree, given the data

Likelihood

- $L(T|D) = P(D|T)$

Likelihood: 10 coin flips → 10 heads
What's the likelihood that the coin is fair?

Probability: Probability of landing heads up 10 times

Likelihood

- $L(T|D) = P(D|T)$
- $L(T|D) = \Pi P(s_i|T)$

Alignment site *i*

Likelihood

- $L(T|D) = P(D|T)$
- $L(T|D) = \Pi\ P(s_i|T)$

Alignment site *i*

What is problematic about this term?

Likelihood

- $L(T|D) = P(D|T)$
- $L(T|D) = \Pi \, P(s_i|T)$
- $\log(L(T|D)) = \Sigma \log(P(s_i|T))$

Likelihood

- $L(T|D) = P(D|T)$
- $L(T|D) = \Pi\ P(s_i|T)$
- $\log(L(T|D)) = \Sigma \log(P(s_i|T))$

This is the model
1. Tree topology
2. Branch lengths
3. Model of nucleotide substitution
   → generally lumped into parameter vector **Θ**: L(**Θ**|D)

Likelihood

- $L(T|D) = P(D|T)$
- $L(T|D) = \Pi\, P(s_i|T)$
- $\log(L(T|D)) = \Sigma \log(P(s_i|T))$

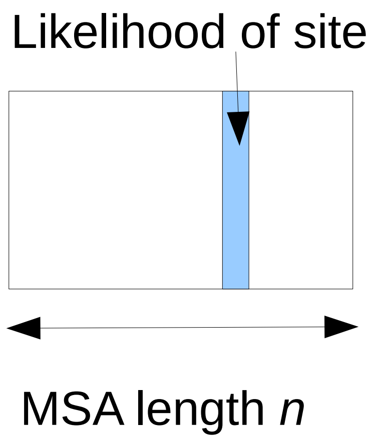
This is the model
1. Tree topology
2. Branch lengths
3. Model of nucleotide substitution

→ generally lumped into parameter vector **Θ**: L(**Θ**|D)

How do we compute this?

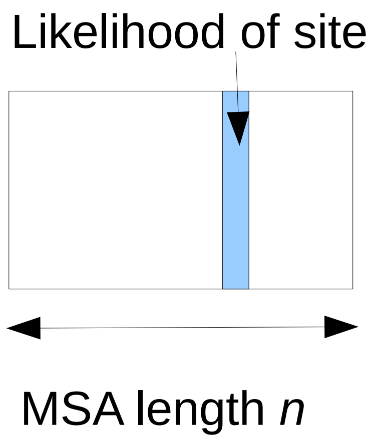
Likelihood of a Tree

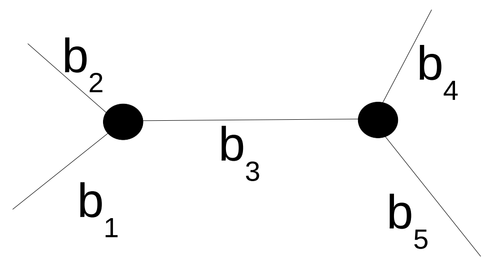
- We assume that sites evolve independently



Likelihood of a Tree

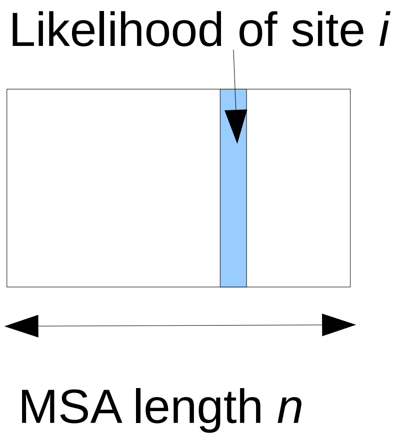
- We assume that sites evolve independently

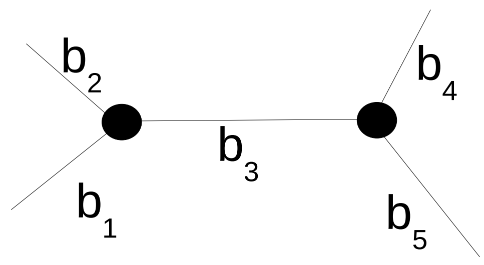




Likelihood of a Tree

- We assume that sites evolve independently



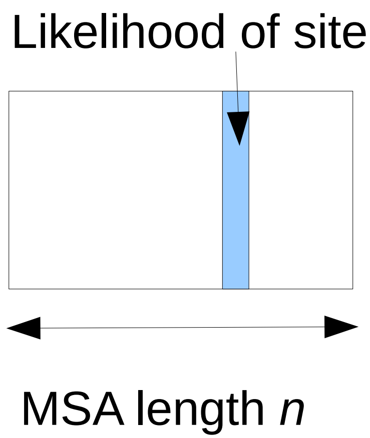


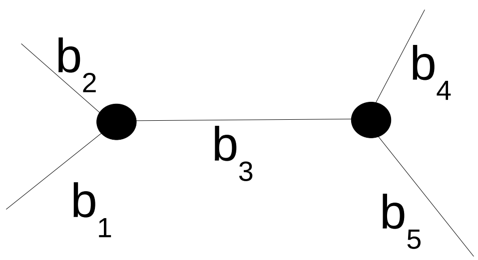
Model ***M***

$P_{ij}(t)$

Likelihood of a Tree

- We assume that sites evolve independently





Model ***M***

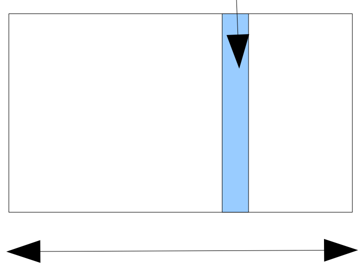
$P_{ij}(t)$

- Overall likelihood: $L := \Pi L_i$

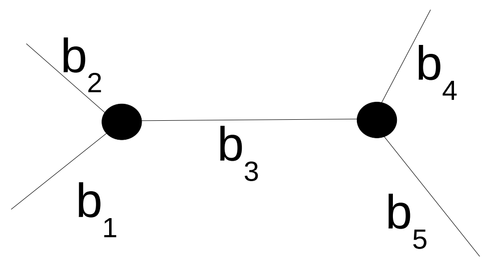
Likelihood of a Tree

- We assume that sites evolve independently

Likelihood of site *i*



MSA length *n*



Model ***M***
$P_{ij}(t)$

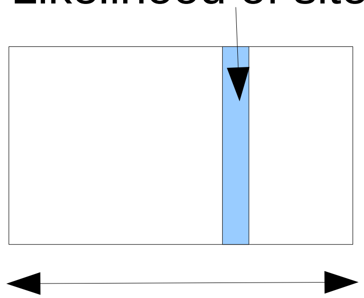
- Overall likelihood: $L := \Pi L_i$
- $P_{ij}(t)$ *i,j in {A, C, G, T}*

Branch length/relative time

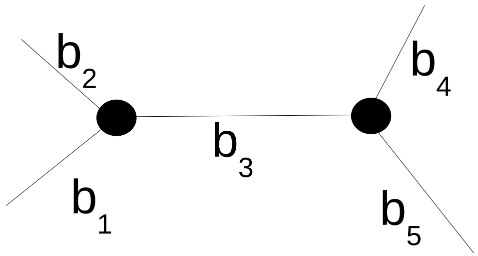
Likelihood of a Tree

- We assume that sites evolve independently

Likelihood of site i



MSA length n



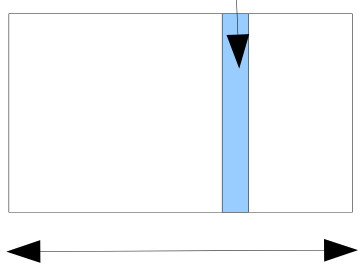
Model ***M***
$P_{ij}(t)$

- Overall likelihood: $L := \Pi L_i$
- $P_{ij}(t)$ *i,j in {A, C, G, T}*
  - → Probability of being in state j after time t
  - → We assume that $P_{ij}(t)$ is a continuous time Markov Process

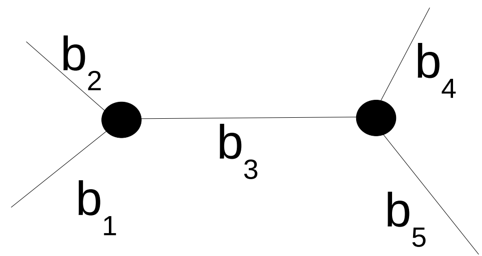
Likelihood of a Tree

- We assume that sites evolve independently

Likelihood of site *i*



MSA length *n*



Model ***M***

$P_{ij}(t)$

- Overall likelihood: $L := \Pi L_i$
- $P_{ij}(t)$ *i,j in {A, C, G, T}*

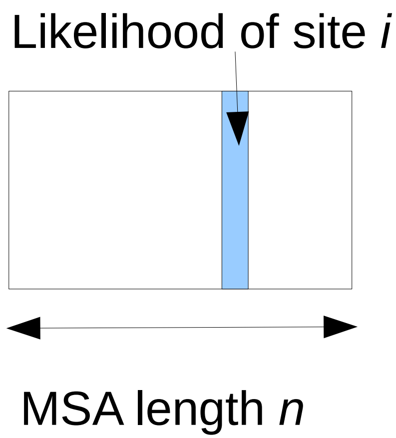
→ Probability of being in state *j* after time *t*

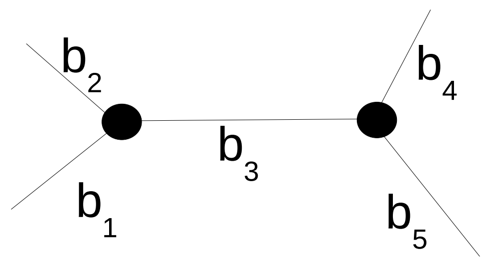
→ We assume that $P_{ij}(t)$ is a Markov Process

- Equilibrium frequency vector $\pi = (\pi_A, \pi_C, \pi_G, \pi_T)$

Likelihood of a Tree

- We assume that sites evolve independently

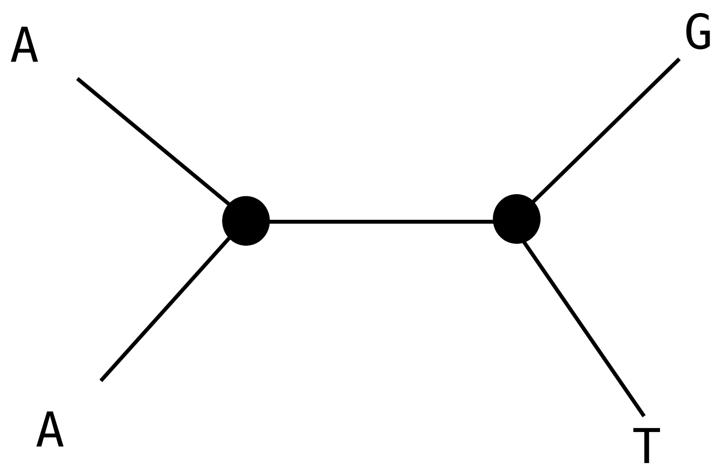




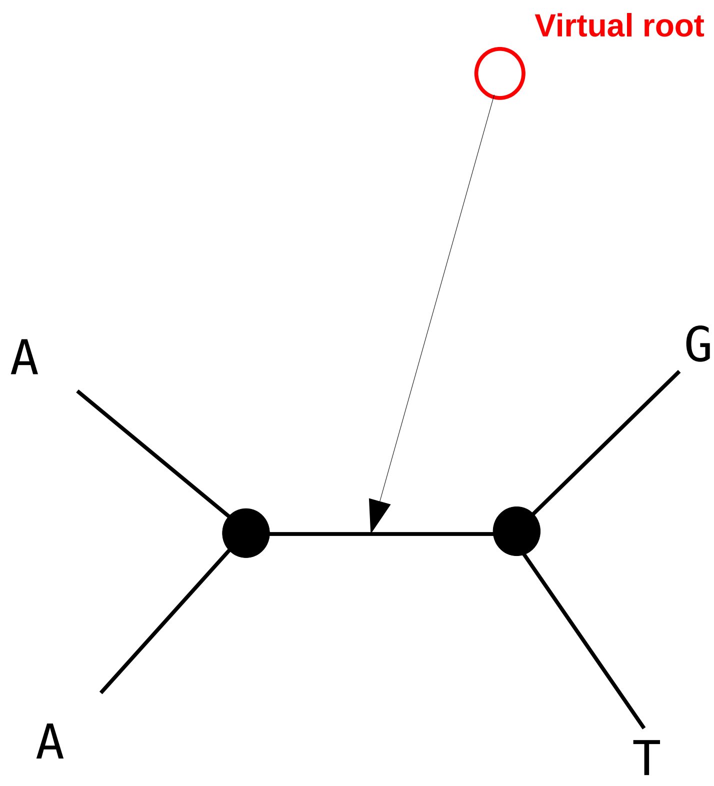
Model ***M***
$P_{ij}(t)$

- Overall likelihood: $L := \Pi L_i$
- $P_{ij}(t)$ *i,j in {A, C, G, T}*
  - → Probability of being in state *j* after time *t*
  - → We assume that $P_{ij}(t)$ is a Markov Process
- Equilibrium frequency vector $\pi = (\pi_A, \pi_C, \pi_G, \pi_T)$
- **Time reversibility:** $\pi_i P_{ij}(t) = \pi_j P_{ji}(t)$

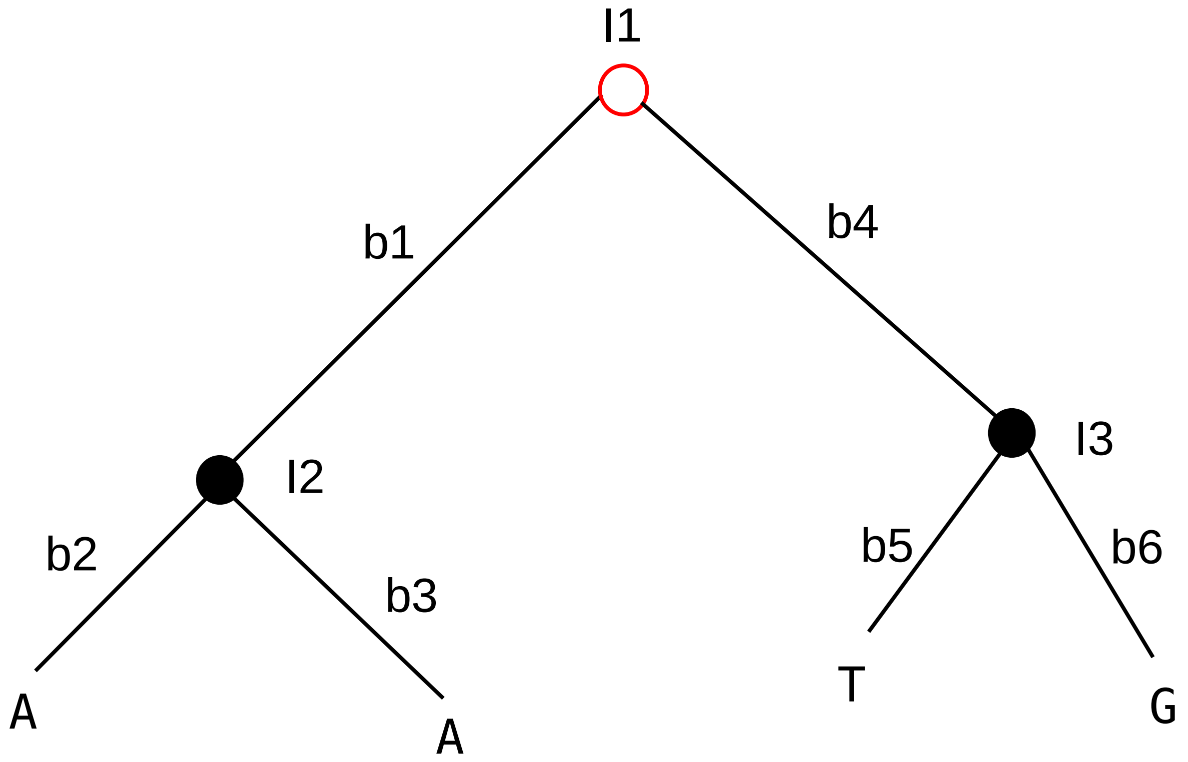
What's the likelihood of this tree?



What's the likelihood of this tree?

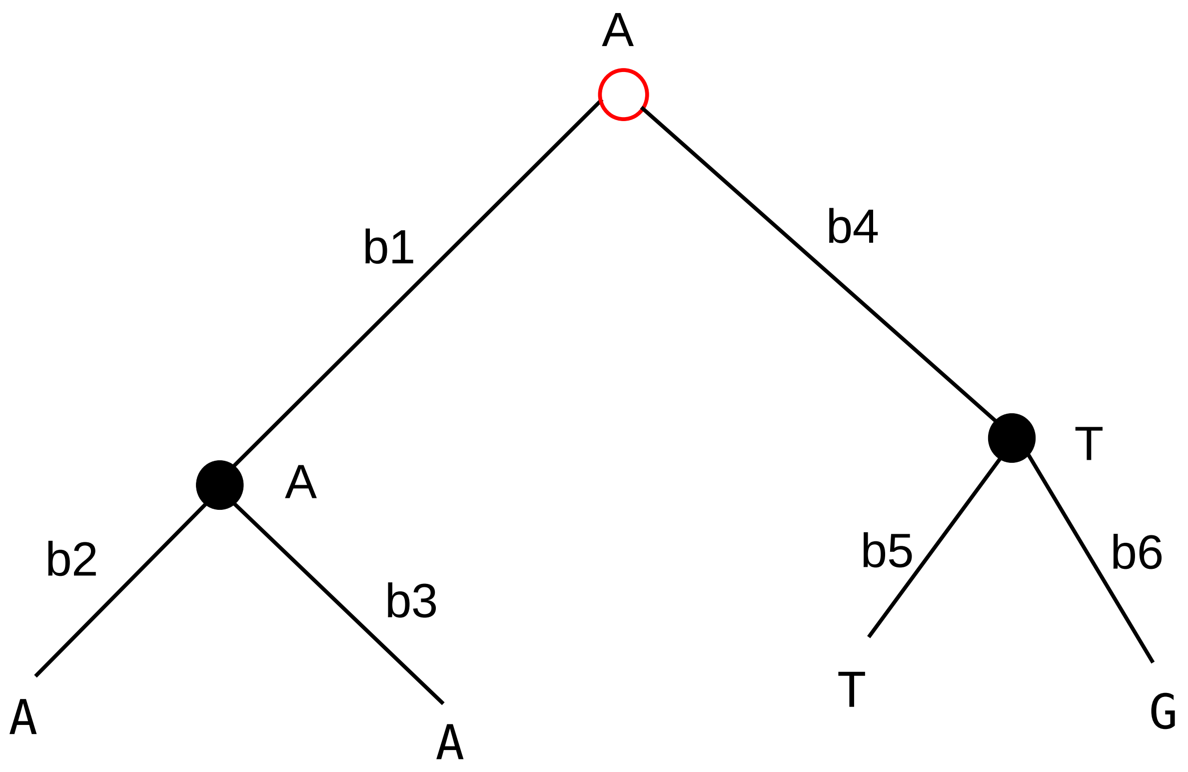


What's the likelihood of this tree?



What's the likelihood of this tree?

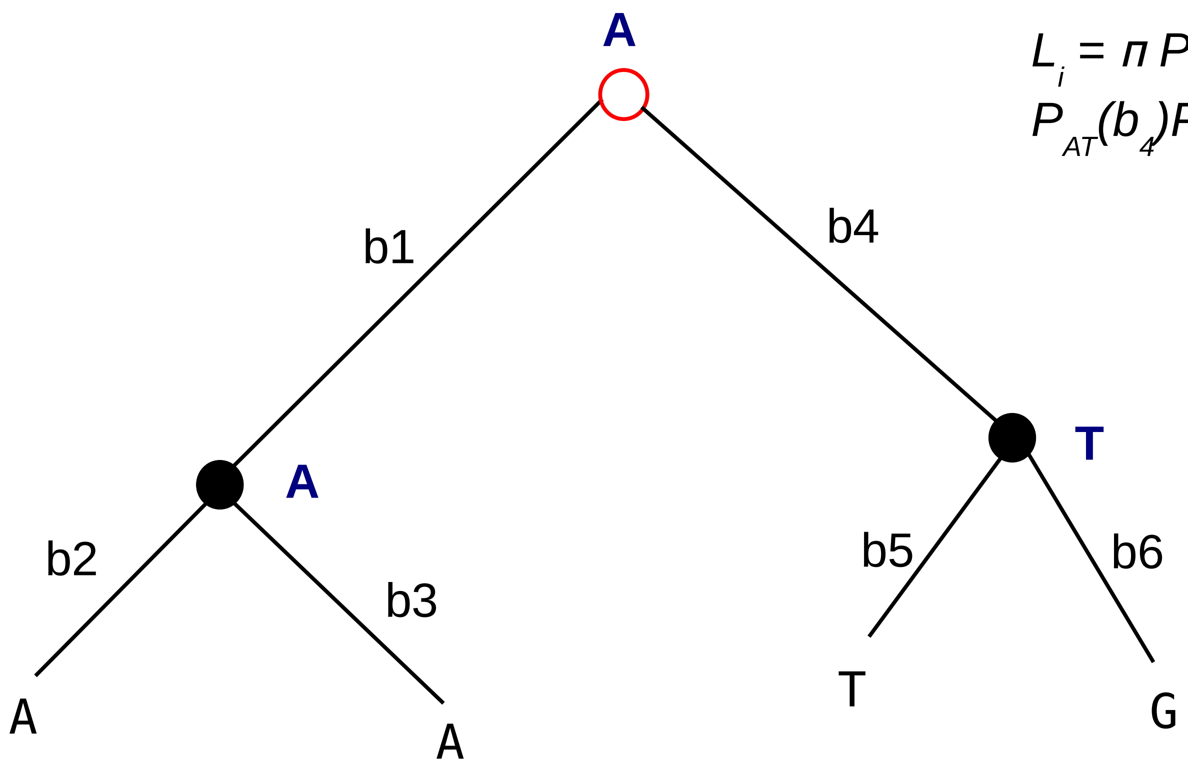
Assume the inner states are given!
What is the likelihood of the tree if we interpret it as **Markov** diagram?



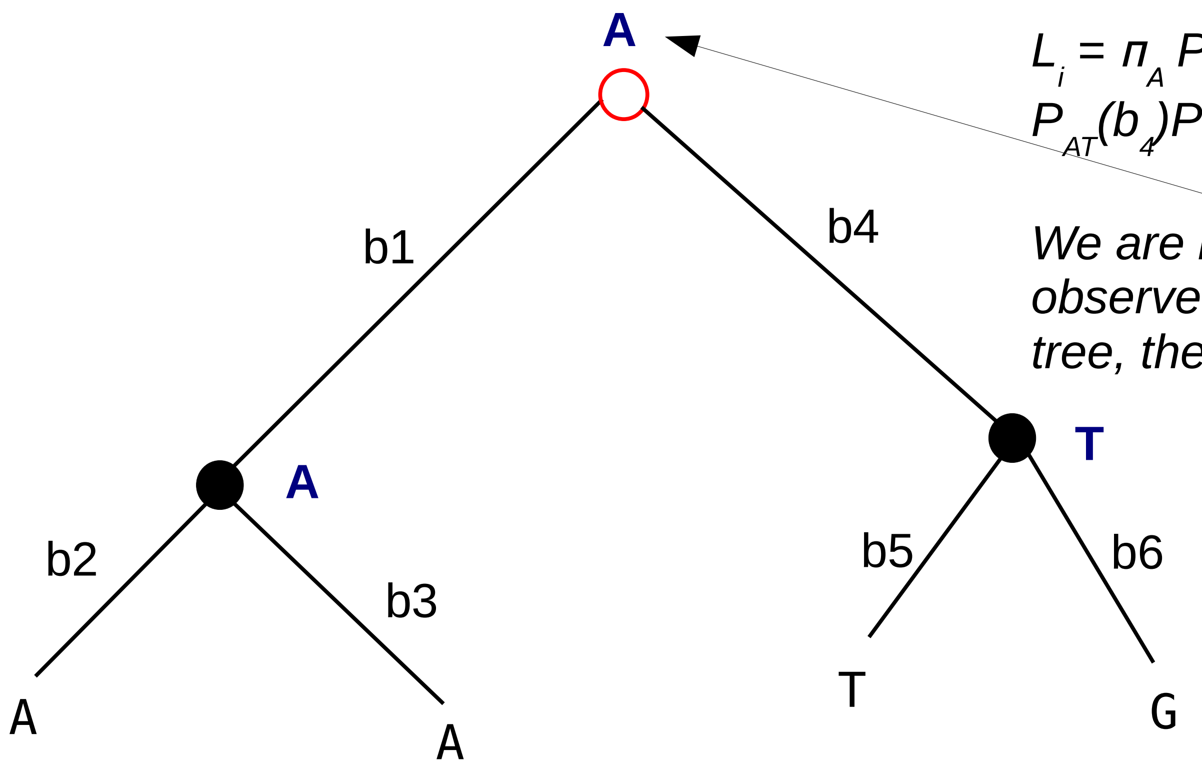
What's the likelihood of this tree?

Assume the inner states are given!
What is the likelihood of the tree if we interpret it as **Markov** diagram?

$$L_i = \pi \, P_{AA}(b_1) \, P_{AA}(b_2) \, P_{AA}(b_3) \, P_{AT}(b_4) P_{TT}(b_5) \, P_{TG}(b_6)$$



What's the likelihood of this tree?

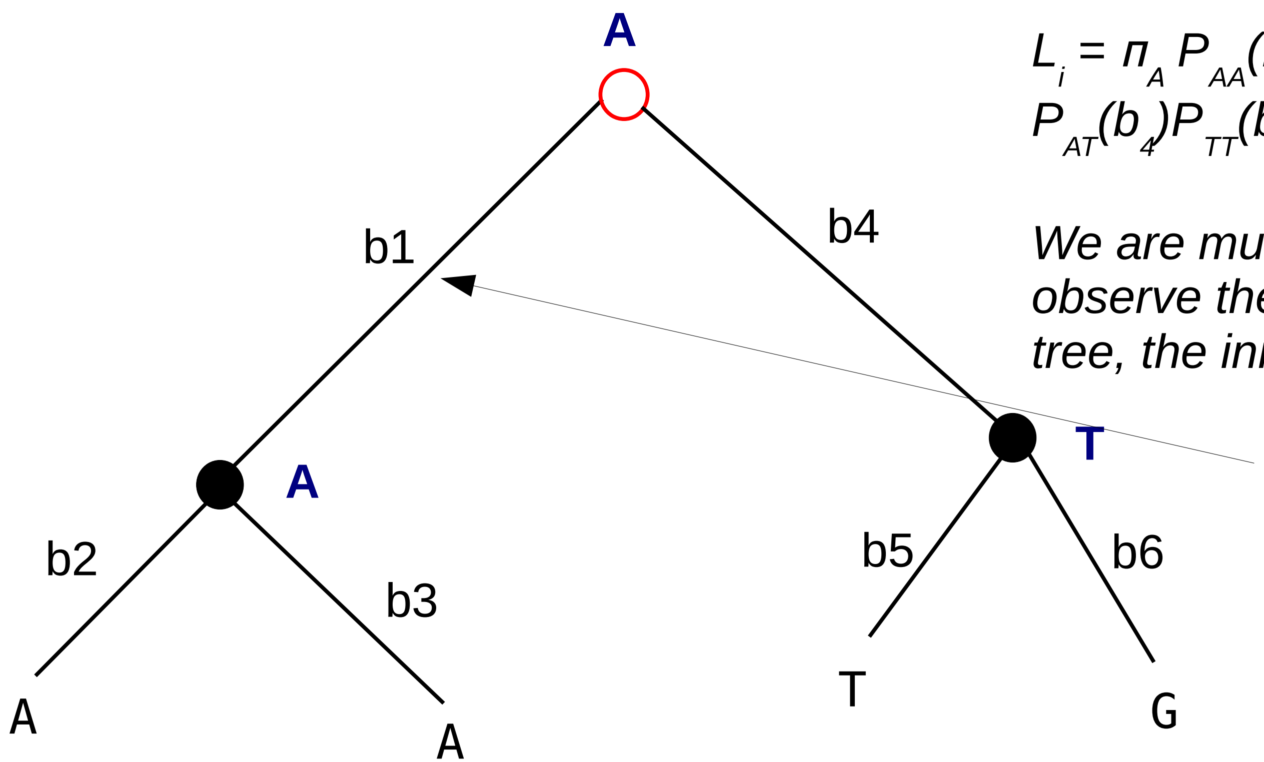


Assume the inner states are given!
What is the likelihood of the tree if we
Interpret it as **Markov** diagram?

$L_i = \pi_A P_{AA}(b_1) P_{AA}(b_2) P_{AA}(b_3) P_{AT}(b_4) P_{TT}(b_5) P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

What's the likelihood of this tree?



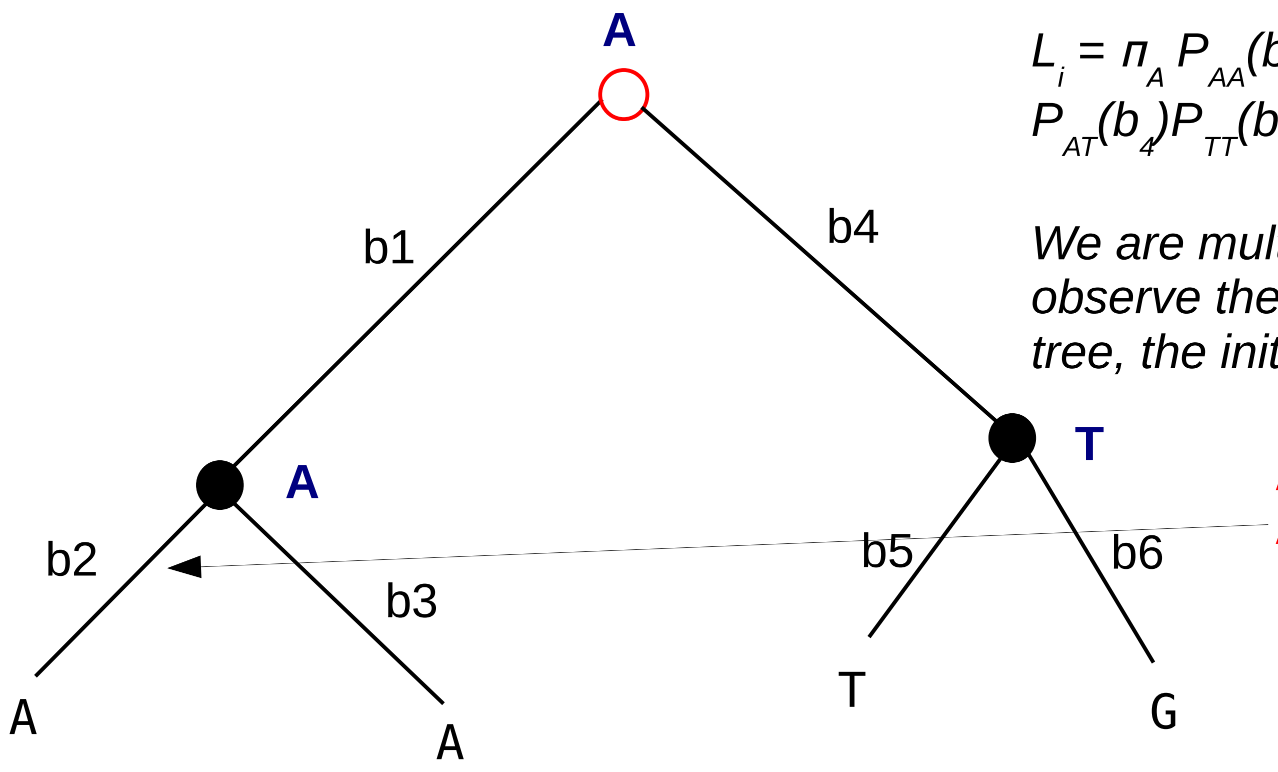
Assume the inner states are given!
What is the likelihood of the tree if we
Interpret it as **Markov** diagram?

$L_i = \pi_A\, P_{AA}(b_1)\, P_{AA}(b_2)\, P_{AA}(b_3)$
$P_{AT}(b_4) P_{TT}(b_5)\, P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened

What's the likelihood of this tree?



Assume the inner states are given!
What is the likelihood of the tree if we
Interpret it as **Markov** diagram?

$L_i = \pi_A\, P_{AA}(b_1)\, P_{AA}(b_2)\, P_{AA}(b_3)$
$P_{AT}(b_4) P_{TT}(b_5)\, P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened
AND this

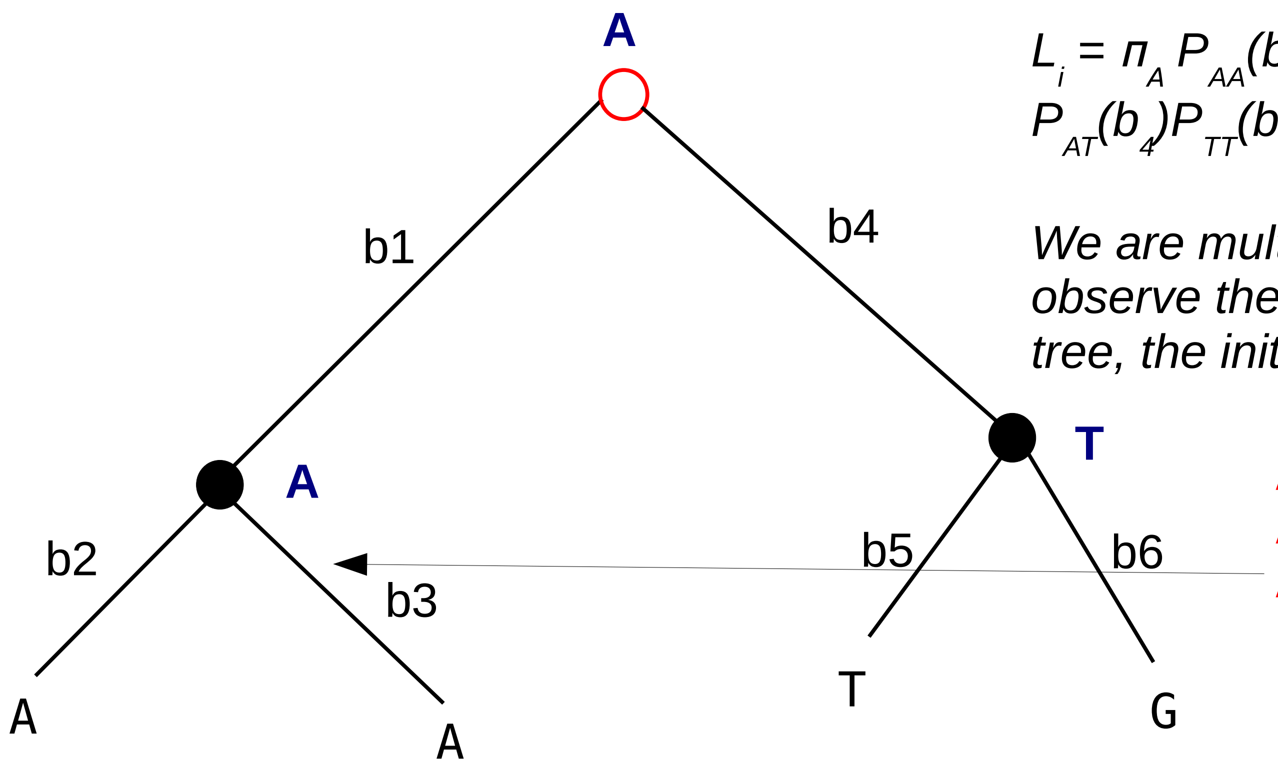
What's the likelihood of this tree?

Assume the inner states are given!
What is the likelihood of the tree if we
Interpret it as **Markov** diagram?

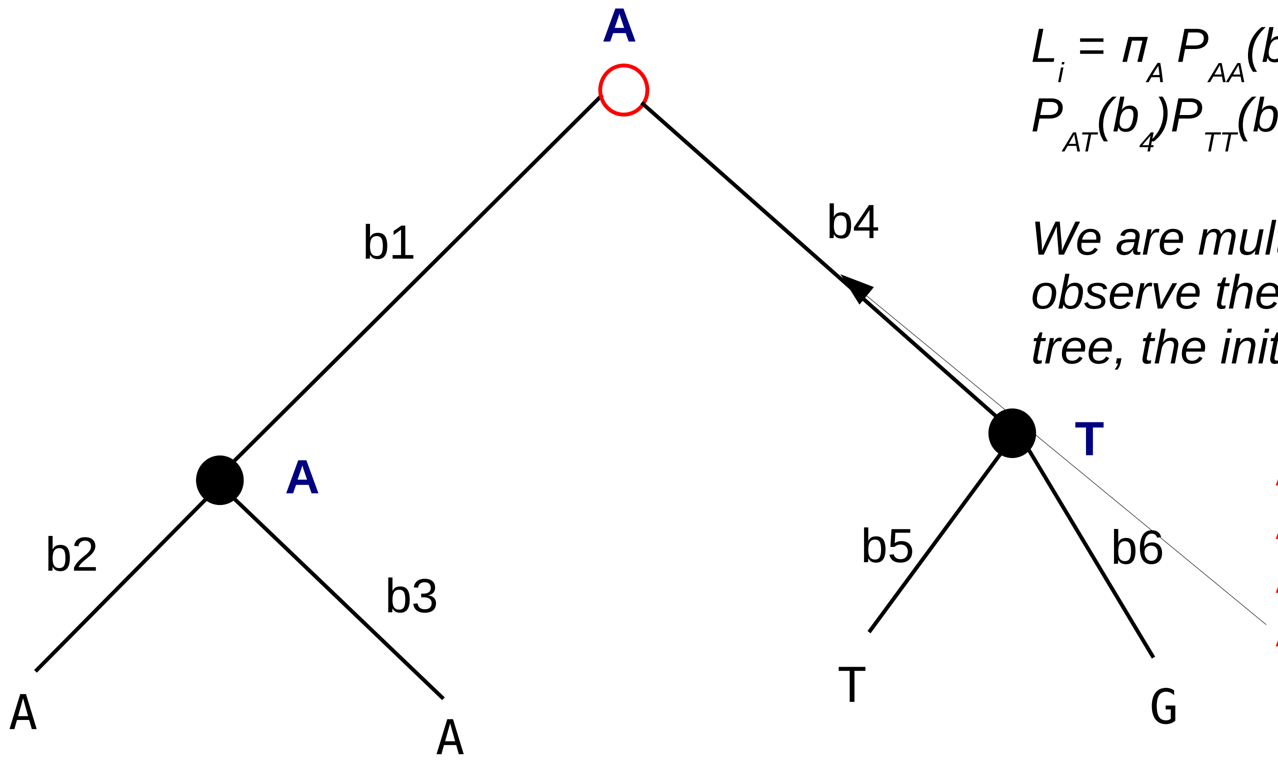
$L_i = \pi_A \, P_{AA}(b_1) \, P_{AA}(b_2) \, P_{AA}(b_3)$
$P_{AT}(b_4) P_{TT}(b_5) \, P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened
AND this
AND this



What's the likelihood of this tree?



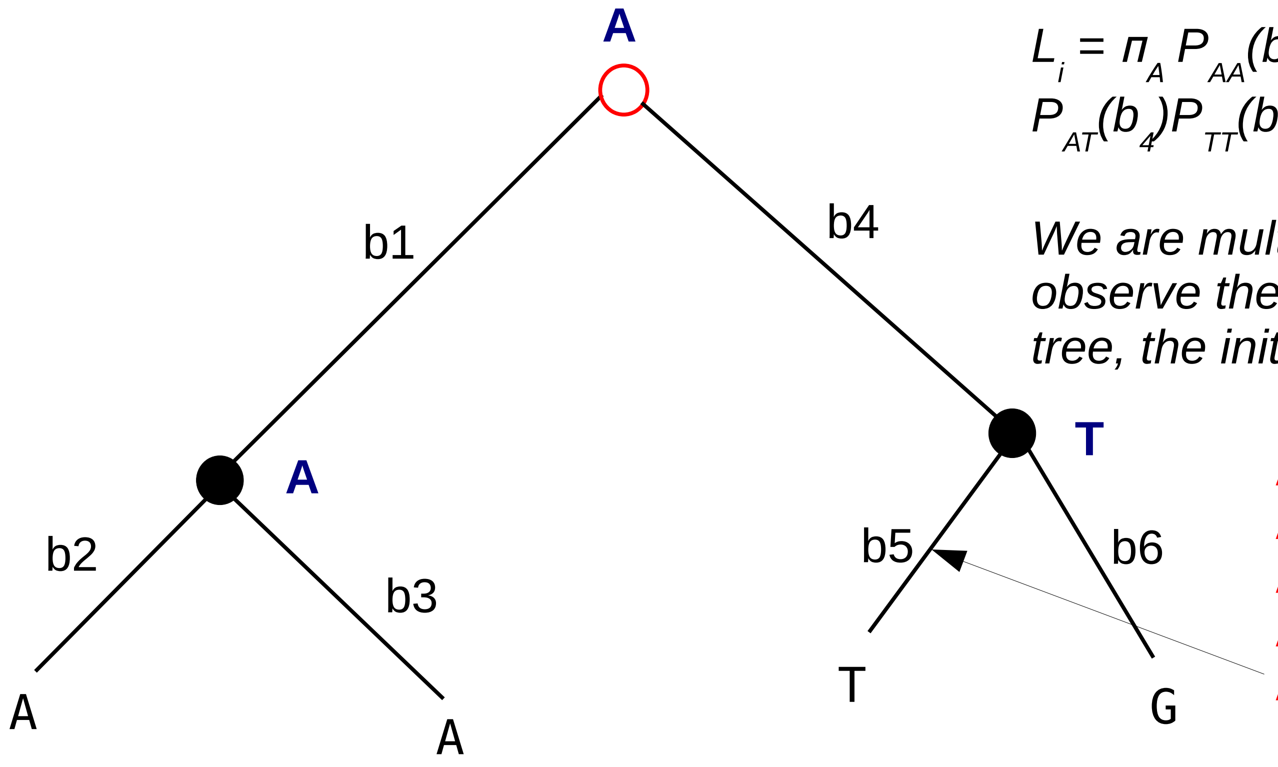
Assume the inner states are given!
What is the likelihood of the tree if we Interpret it as **Markov** diagram?

$L_i = \pi_A P_{AA}(b_1) P_{AA}(b_2) P_{AA}(b_3) P_{AT}(b_4) P_{TT}(b_5) P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened
AND this
AND this
AND this

What's the likelihood of this tree?



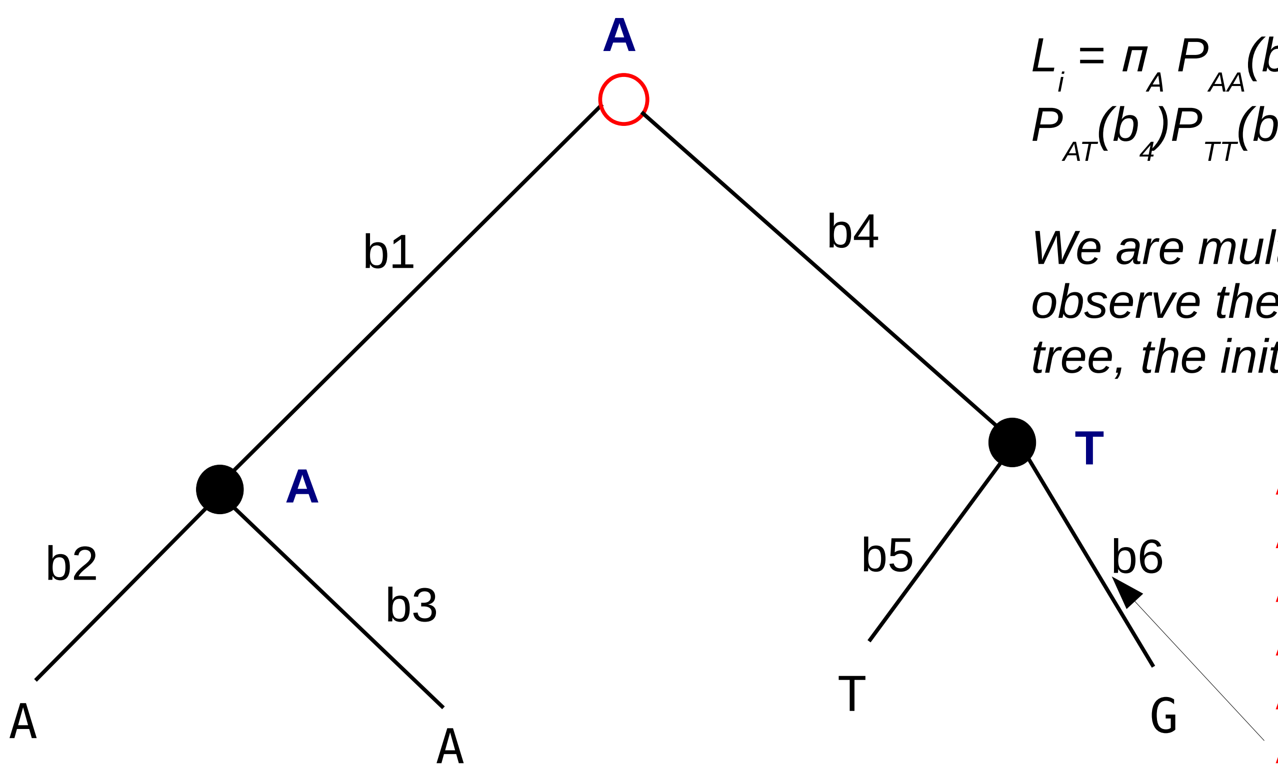
Assume the inner states are given!
What is the likelihood of the tree if we
Interpret it as **Markov** diagram?

$L_i = \pi_A P_{AA}(b_1) P_{AA}(b_2) P_{AA}(b_3) P_{AT}(b_4) P_{TT}(b_5) P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened
AND this
AND this
AND this
AND this

What's the likelihood of this tree?



Assume the inner states are given!
What is the likelihood of the tree if we Interpret it as **Markov** diagram?

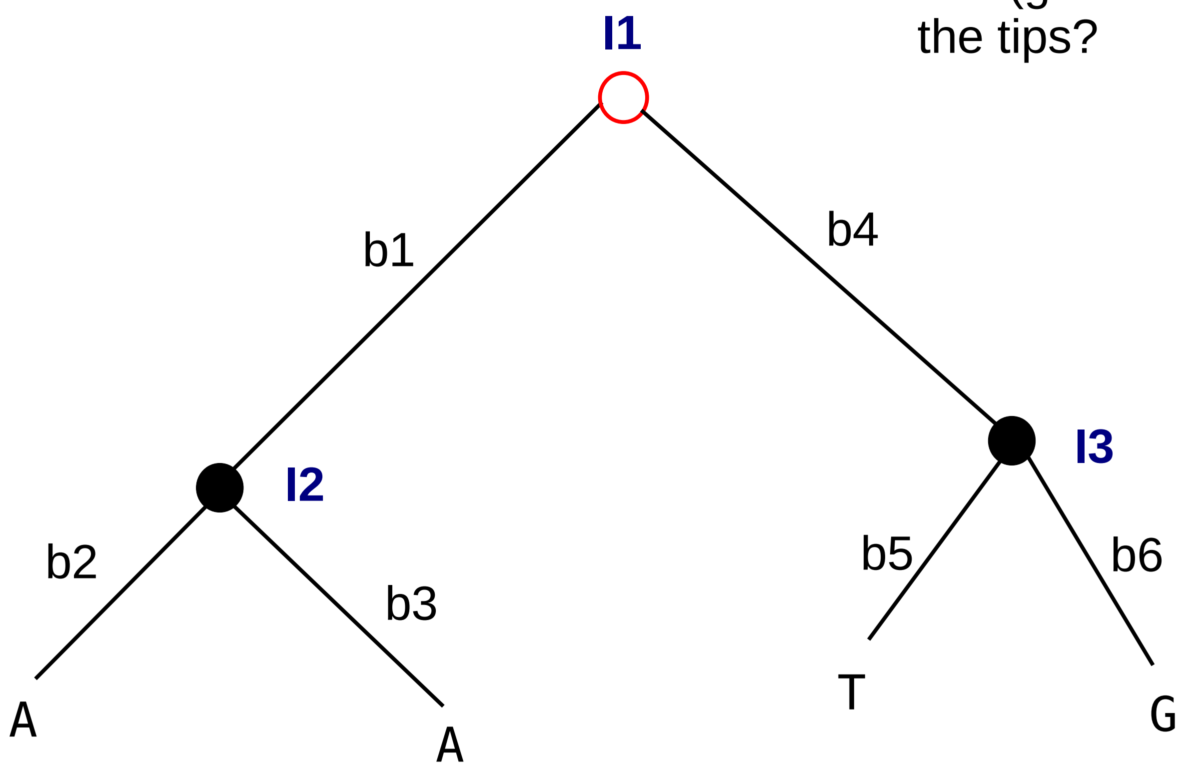
$L_i = \pi_A P_{AA}(b_1) P_{AA}(b_2) P_{AA}(b_3) P_{AT}(b_4) P_{TT}(b_5) P_{TG}(b_6)$

We are multiplying here, because to observe the data at the tips, given the tree, the initial state must be **A** π_A

AND then this happened
AND this
AND this
AND this
AND this
AND this

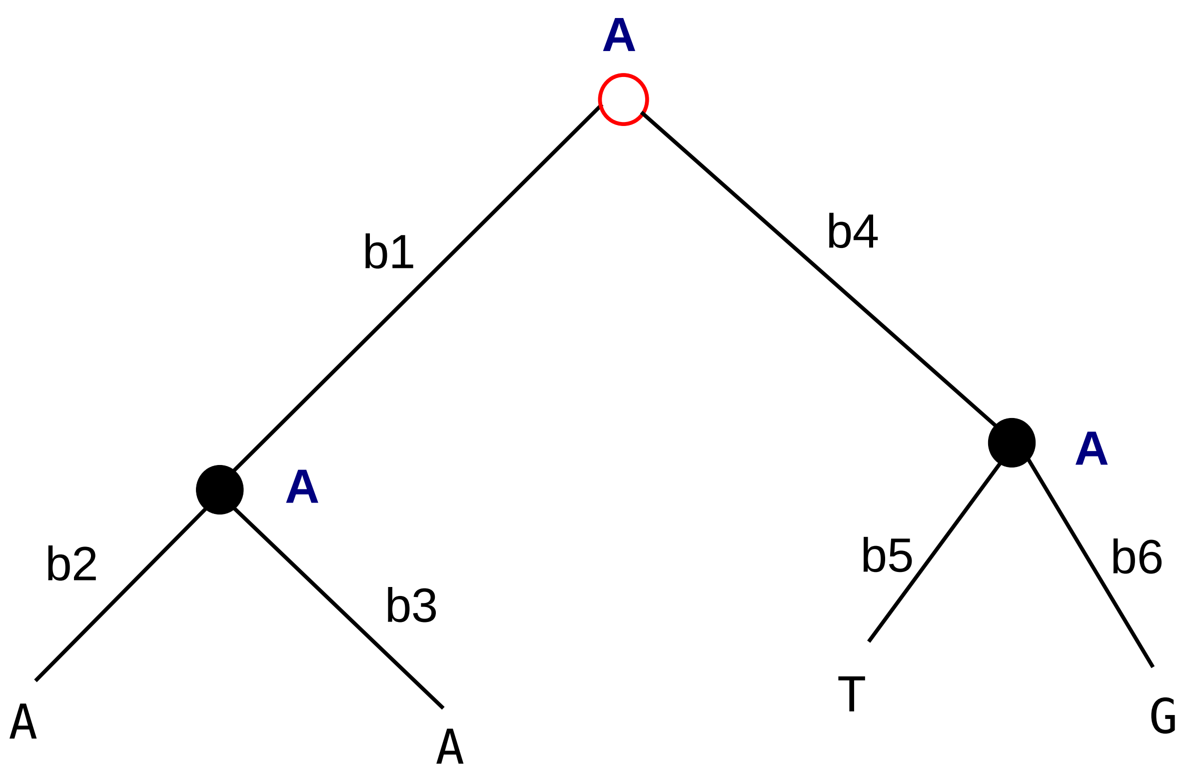
What's the likelihood of this tree?

However, we don't know the inner states :-(
So the question is: What are the possible evolutionary histories that could have given rise (generated) to the data we observe at the tips?



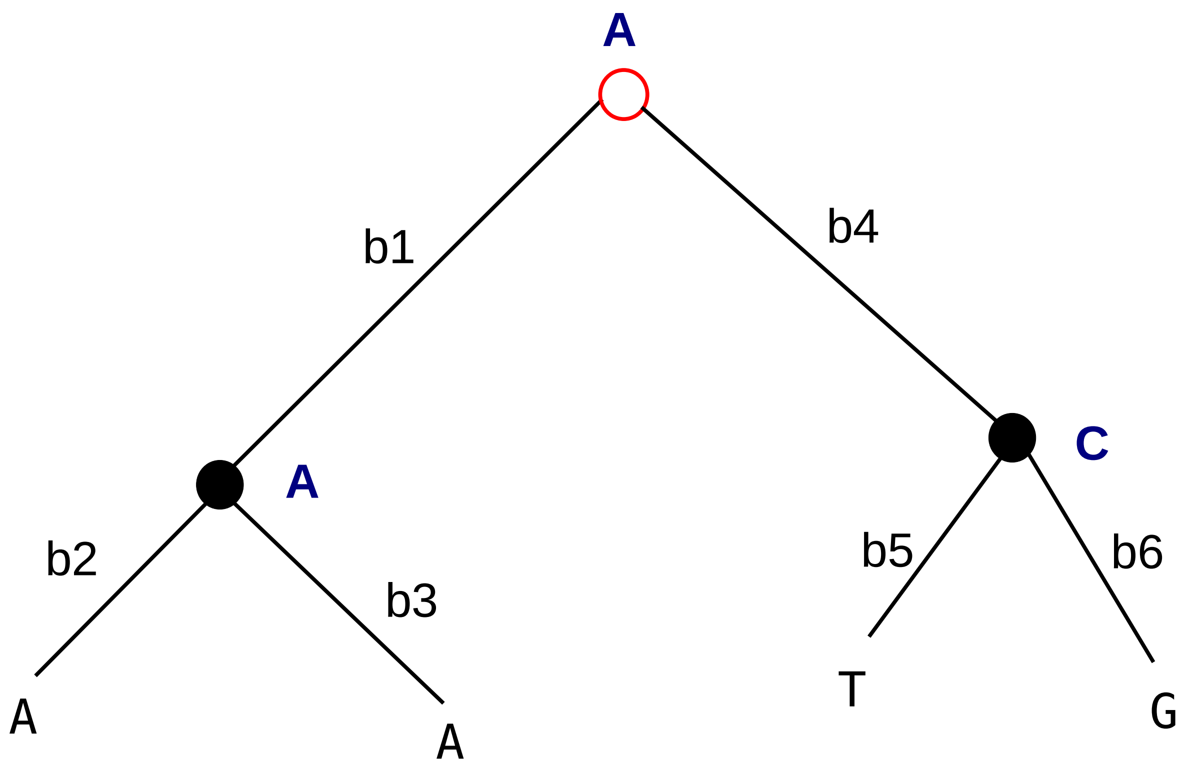
What's the likelihood of this tree?

It could be this



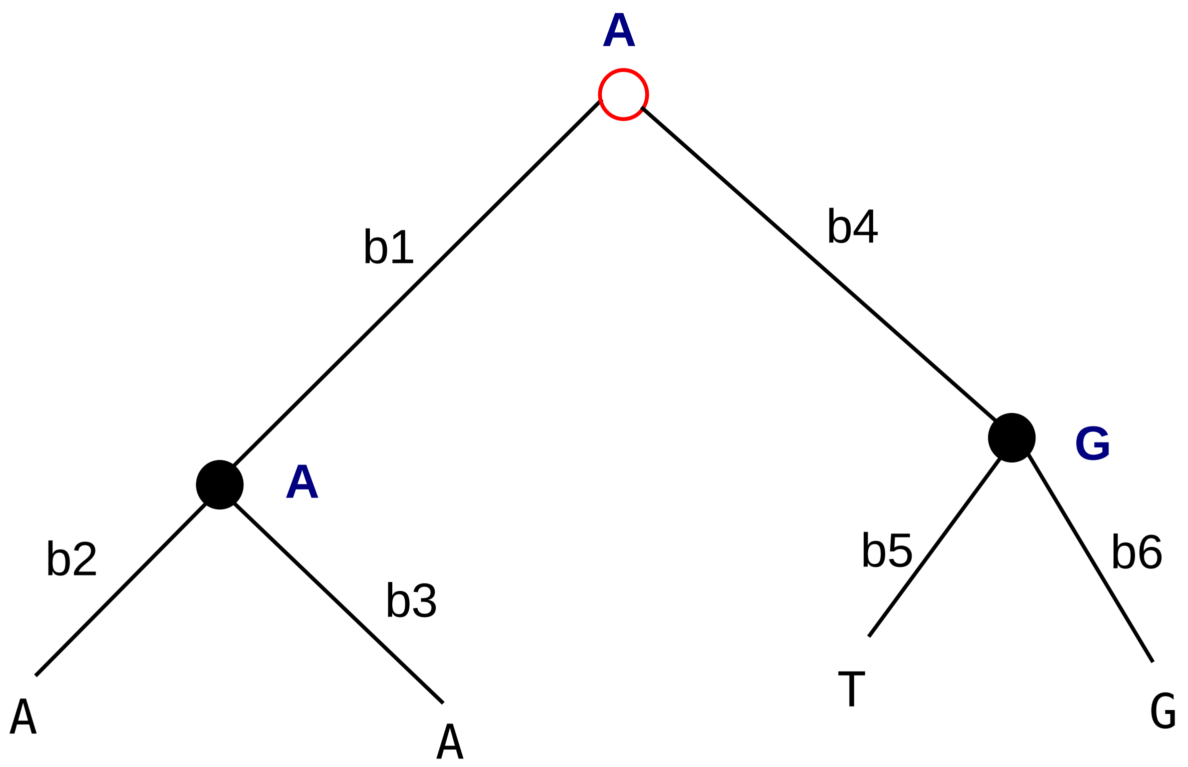
What's the likelihood of this tree?

It could be this
OR this



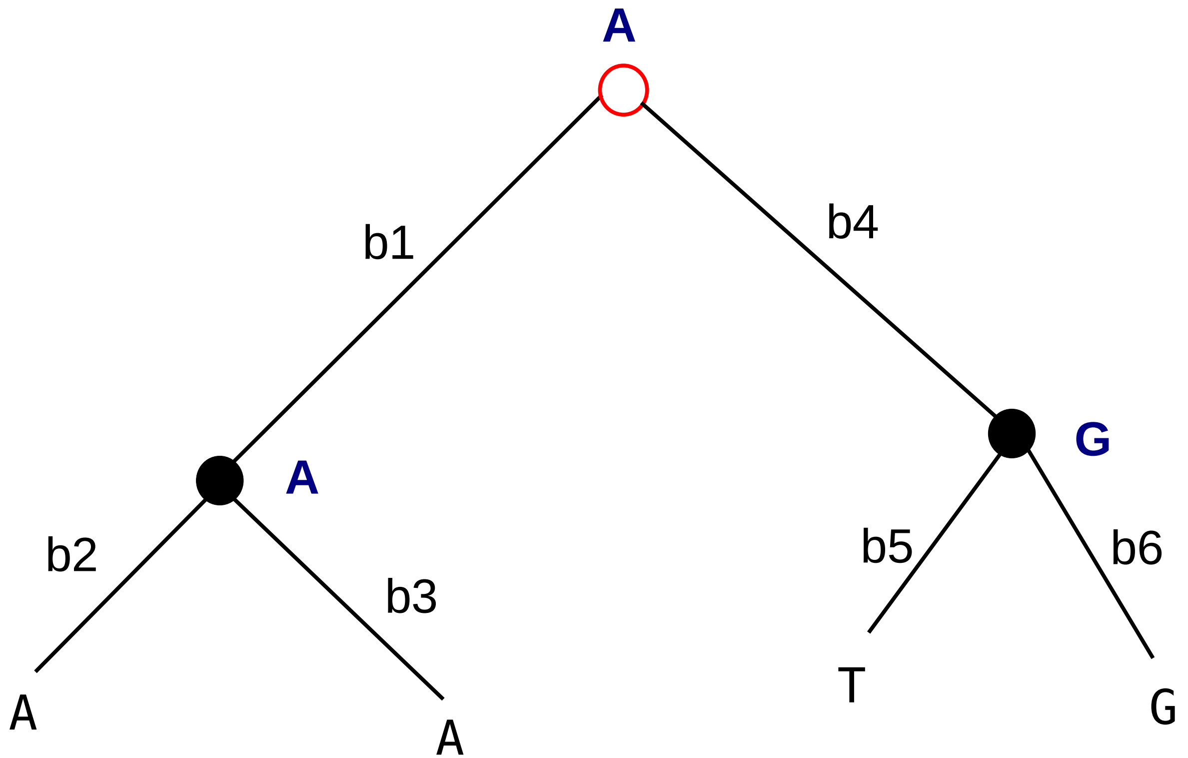
What's the likelihood of this tree?

It could be this
OR this
OR this

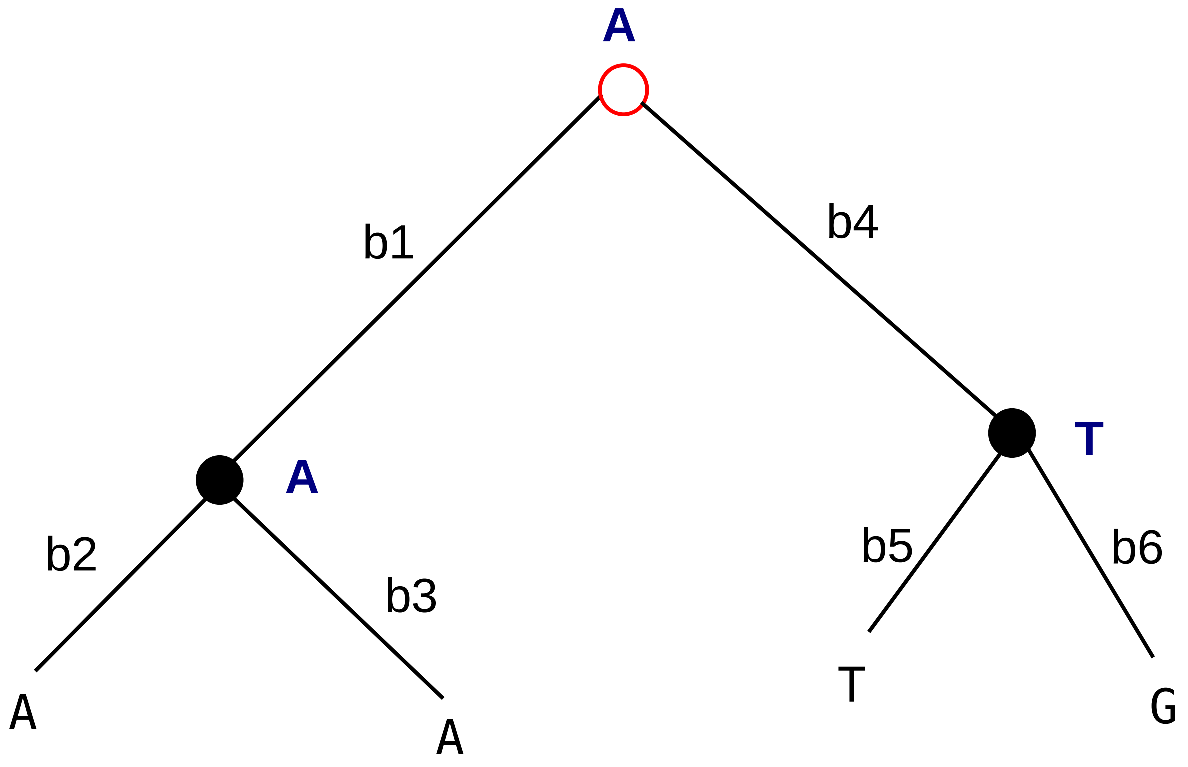


What's the likelihood of this tree?

It could be this
OR this
OR this
OR this

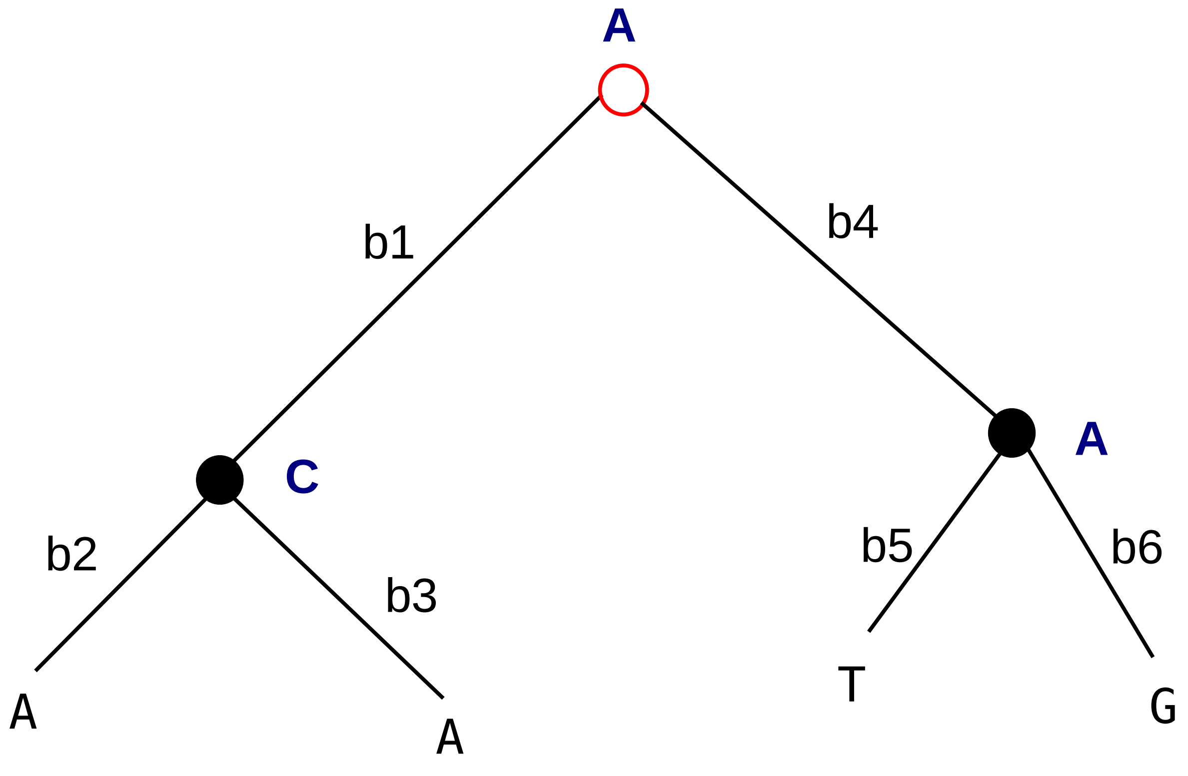


What's the likelihood of this tree?



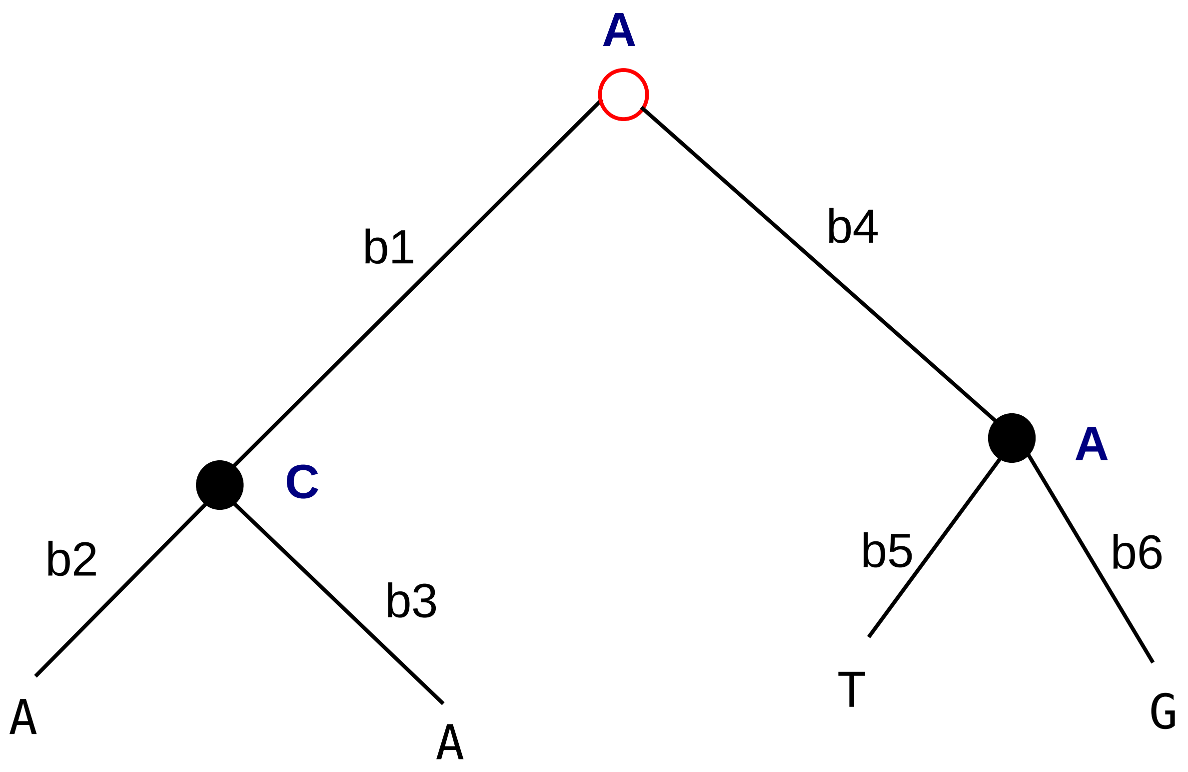
It could be this
OR this
OR this
OR this
OR this

What's the likelihood of this tree?



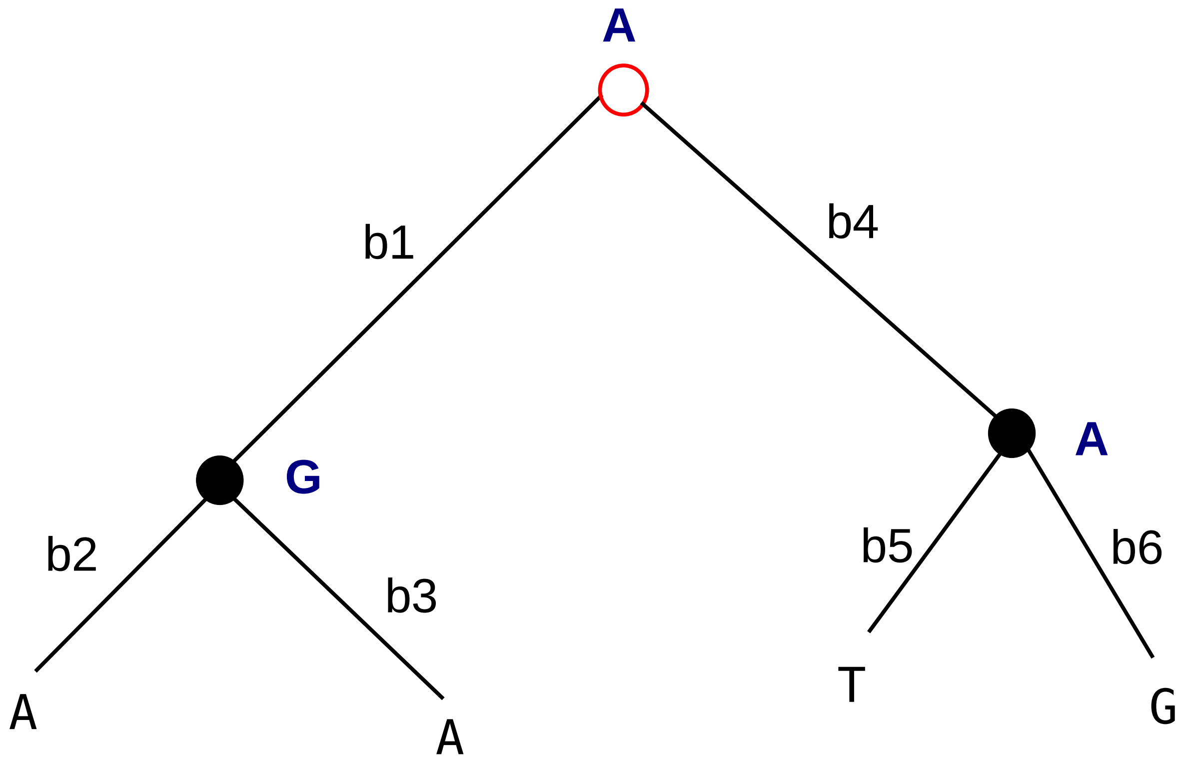
It could be this
OR this
OR this
OR this
OR this
OR this

What's the likelihood of this tree?



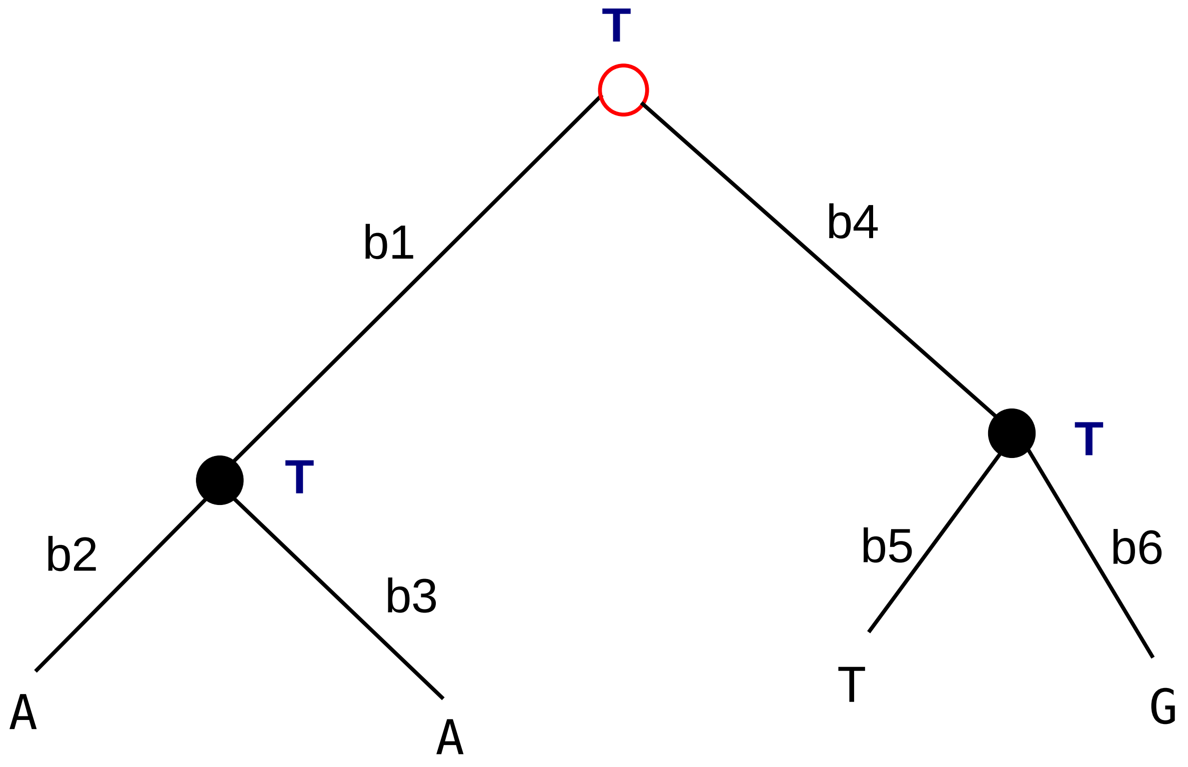
It could be this
OR this
OR this
OR this
OR this
OR this
OR this

What's the likelihood of this tree?



It could be this
OR this
OR this
OR this
OR this
OR this
OR this
OR this

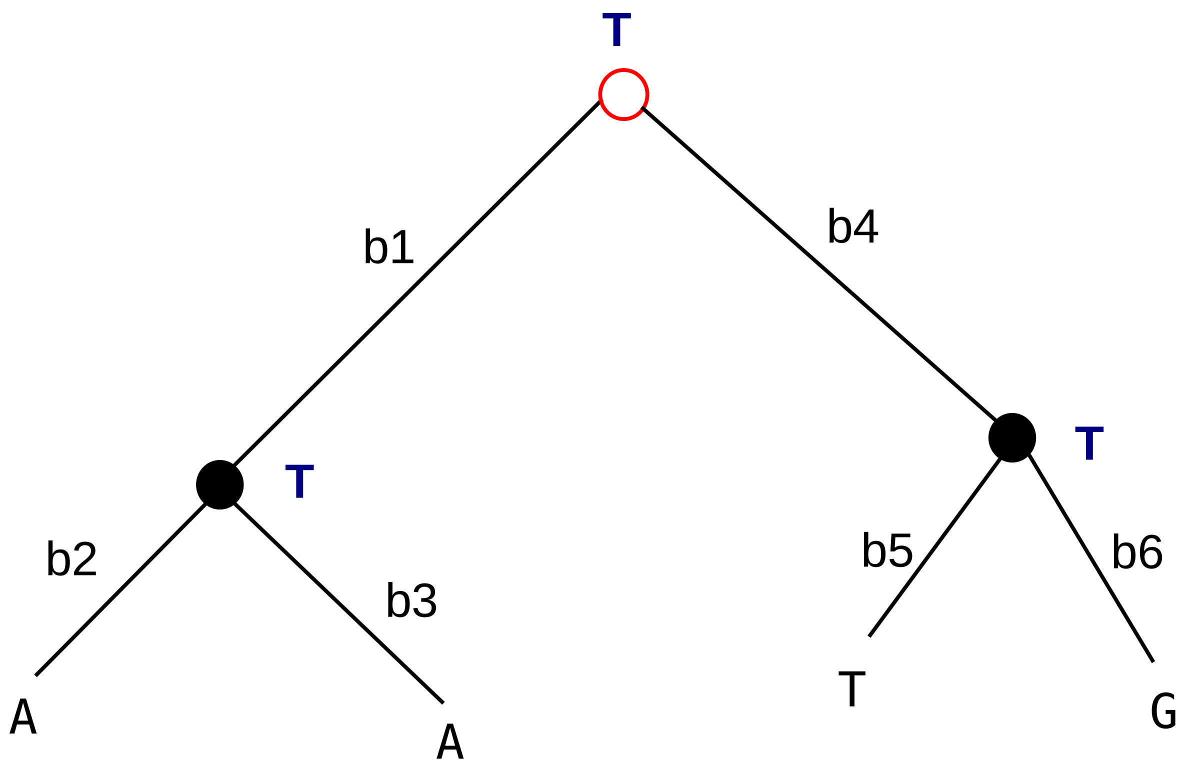
What's the likelihood of this tree?



It could be this
OR this
OR this
OR this
OR this
OR this
OR this
OR this
…
OR this

What's the likelihood of this tree?

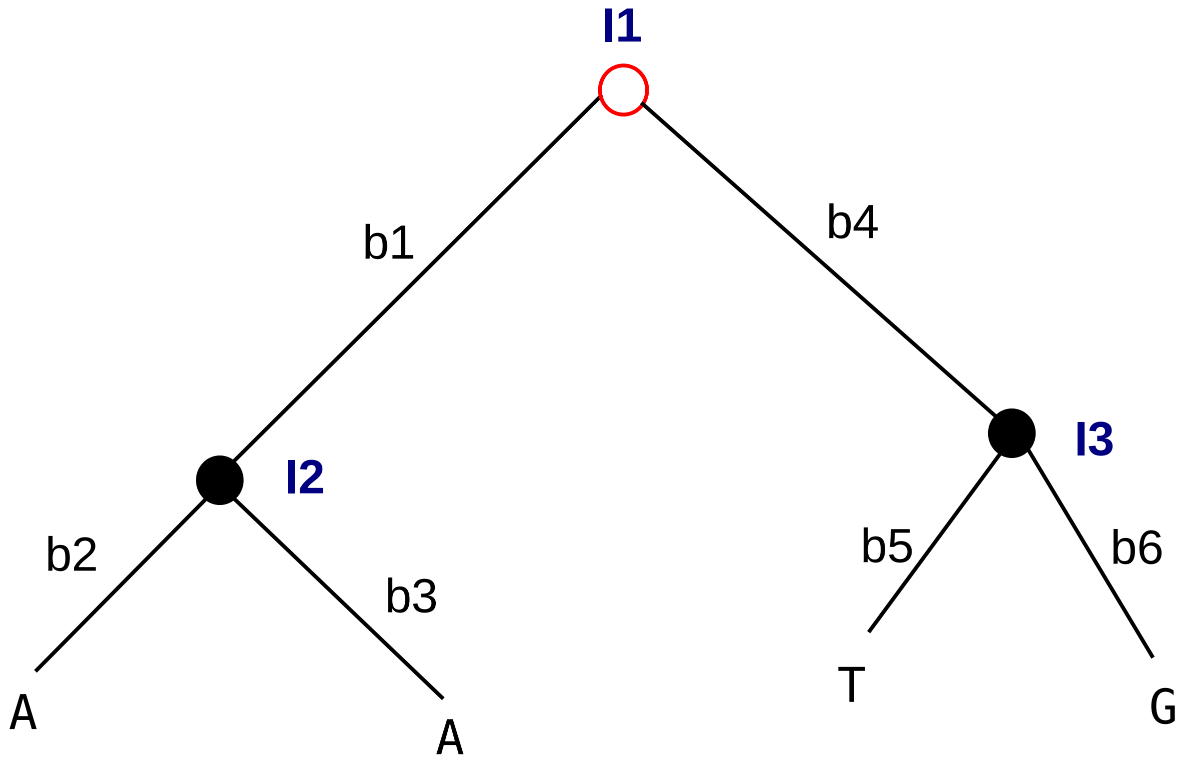
So the likelihood of the tree is the sum (**OR!**) over the likelihoods of all possible assignments of A, C, G, and T (all possible evolutionary histories) to the inner nodes *I1, I2, I3* of the tree.



What's the likelihood of this tree?

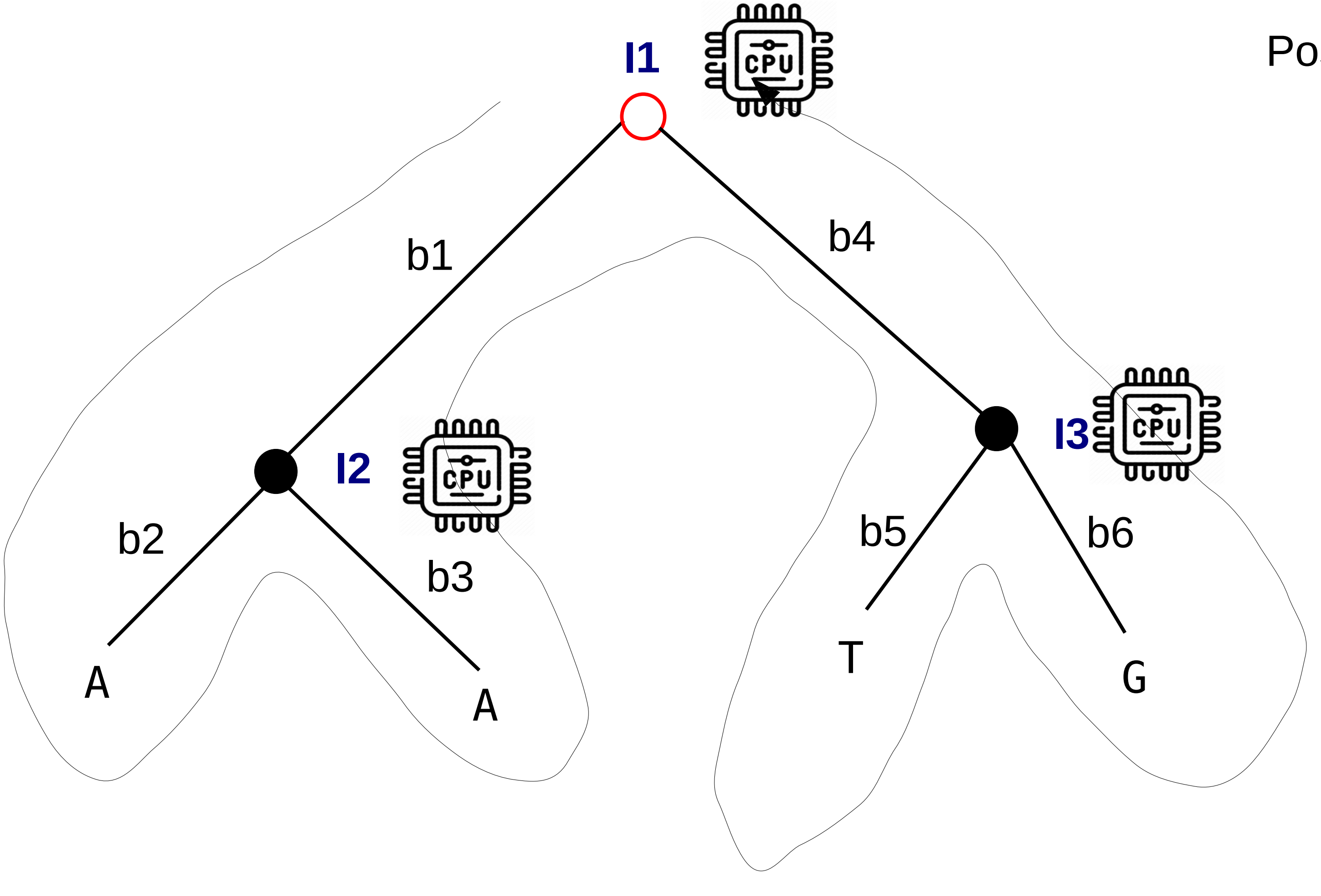
So the likelihood of the tree is the sum (**OR!**) over the likelihoods of all possible assignments of A, C, G, and T (all possible evolutionary histories) to the inner nodes *I1, I2, I3* of the tree.

There are 4 x 4 x 4 possible assignments in our example
→ this sounds very compute-intensive :-(

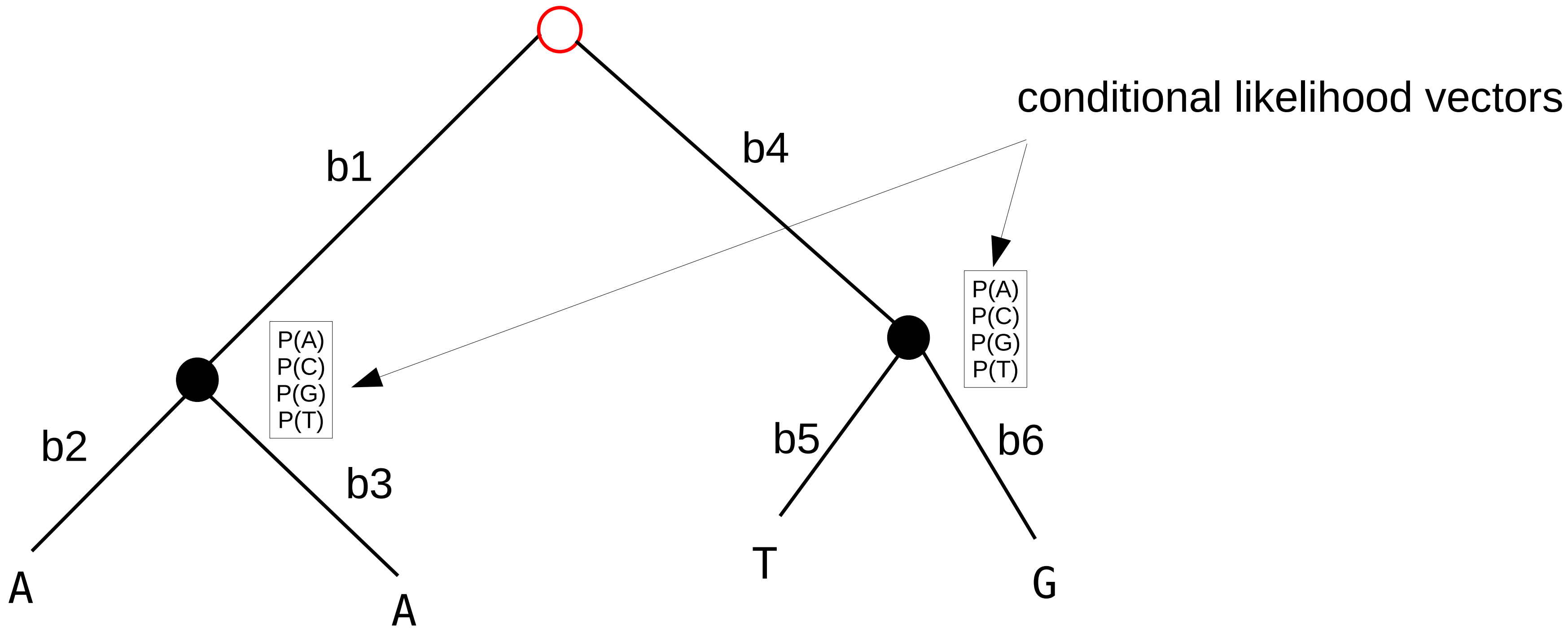


The Felsenstein Pruning Algorithm

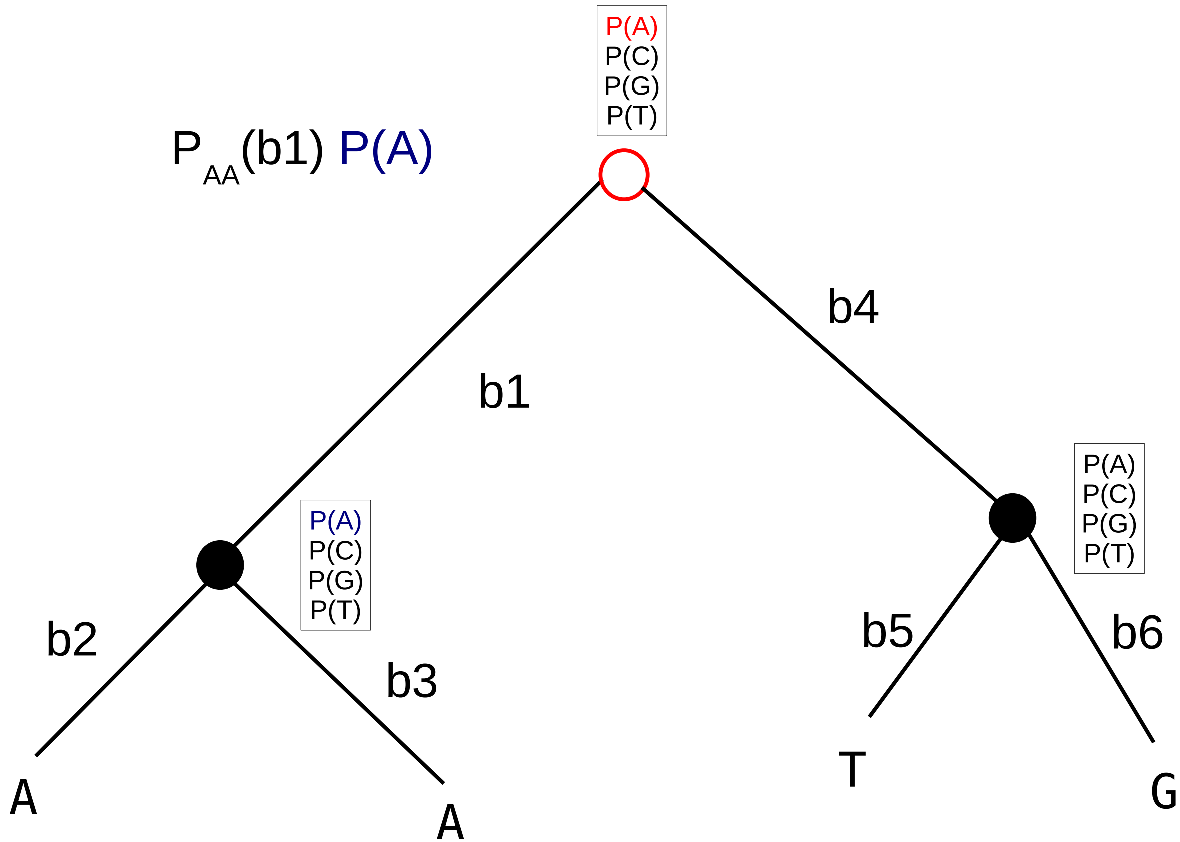
Post order traversal



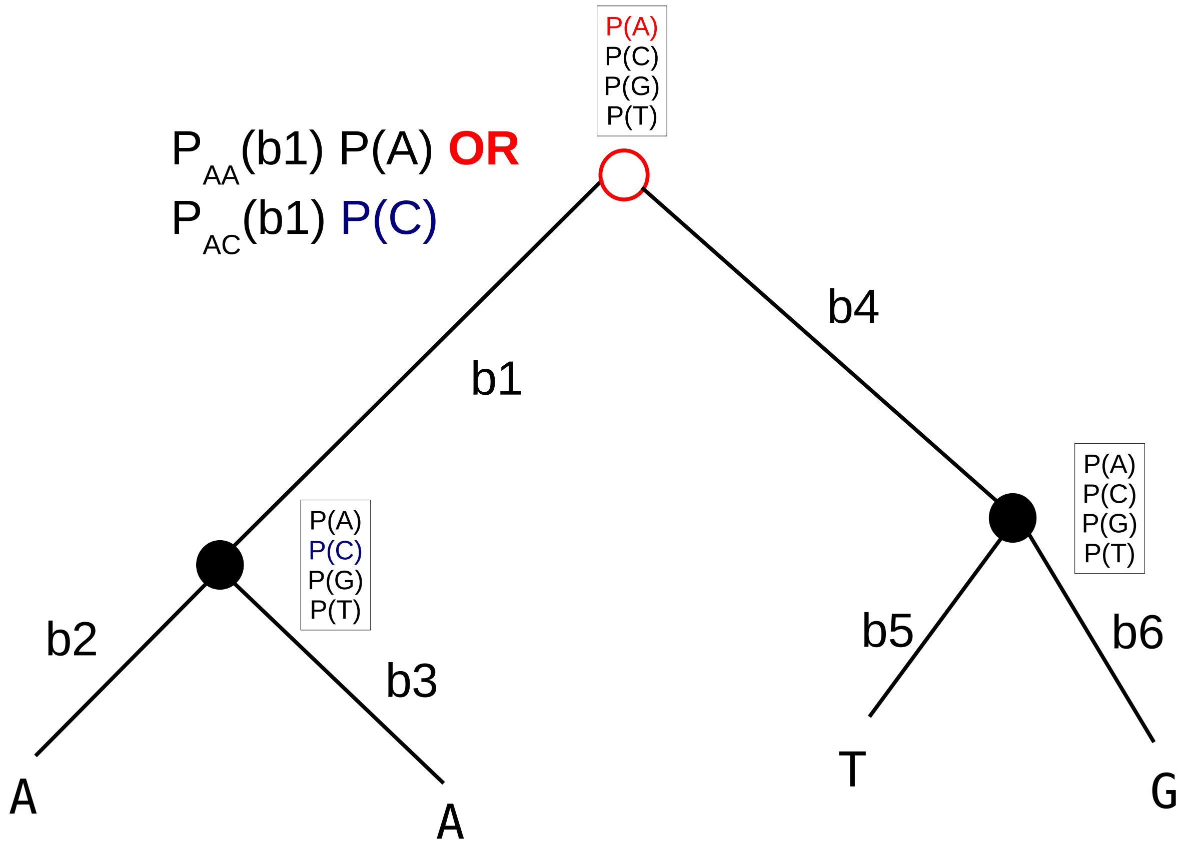
Felsenstein Pruning



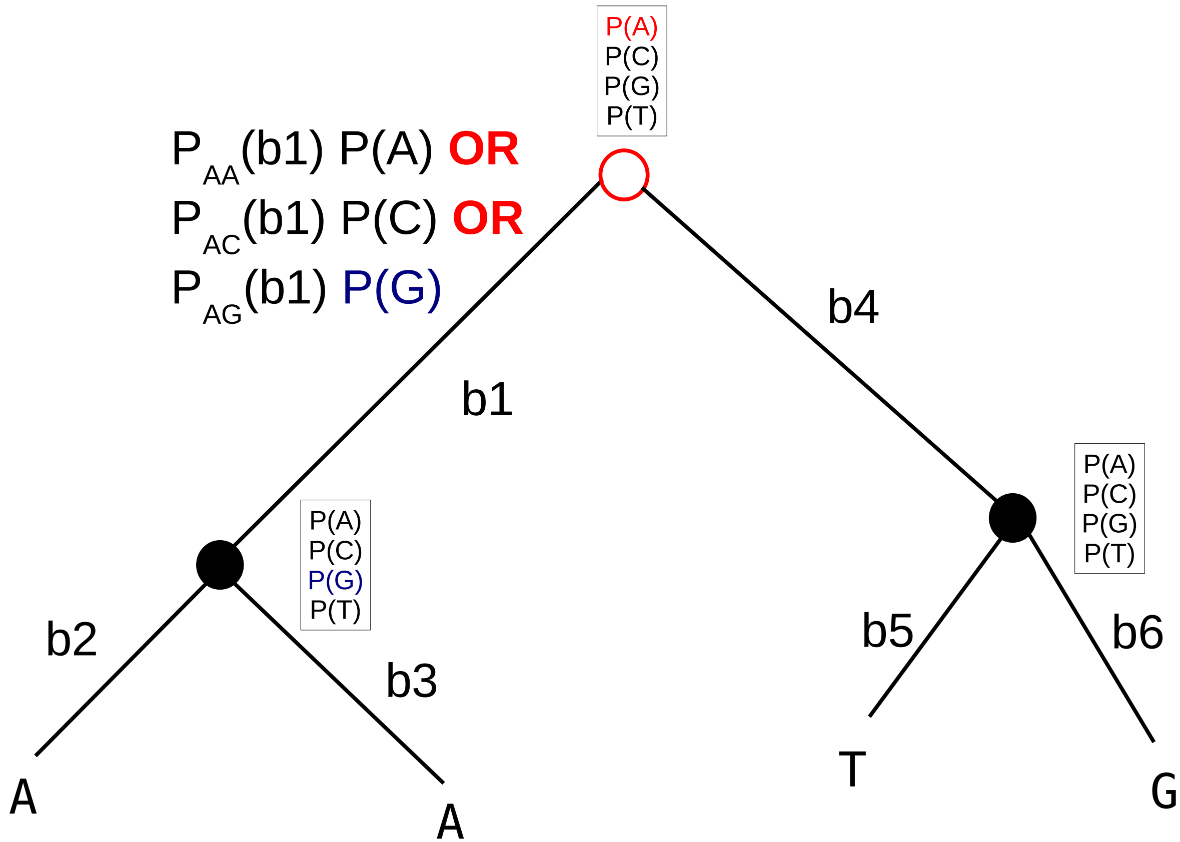
Felsenstein Pruning



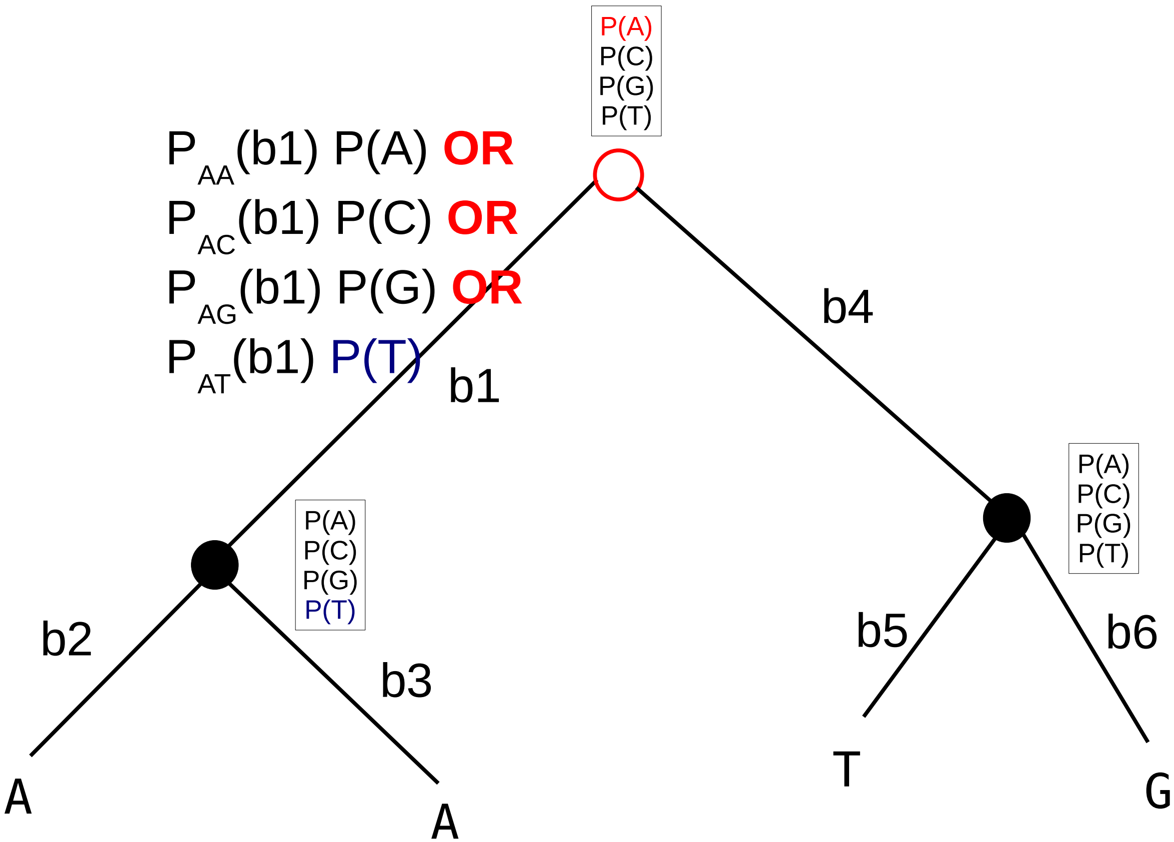
Felsenstein Pruning



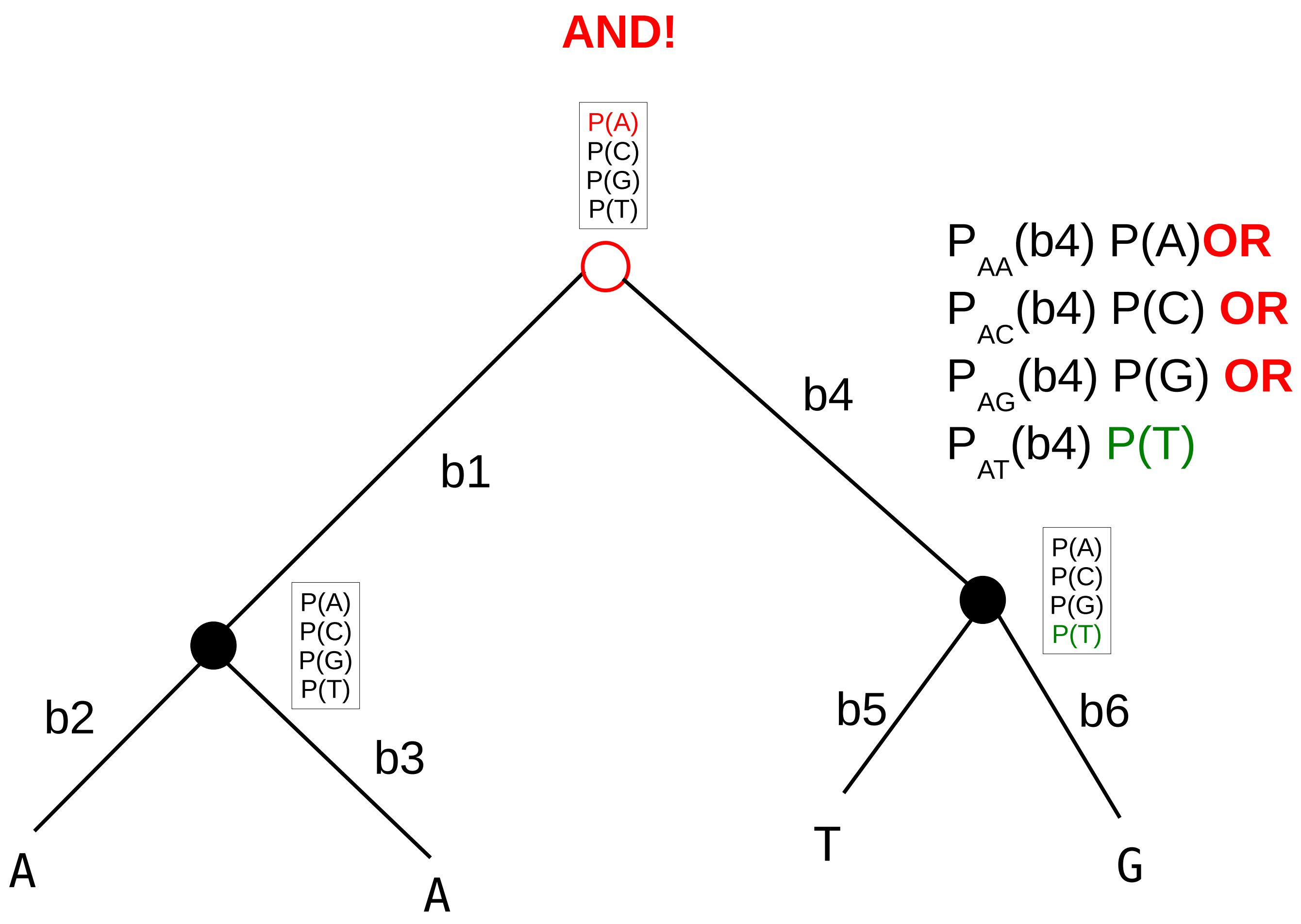
Felsenstein Pruning



Felsenstein Pruning



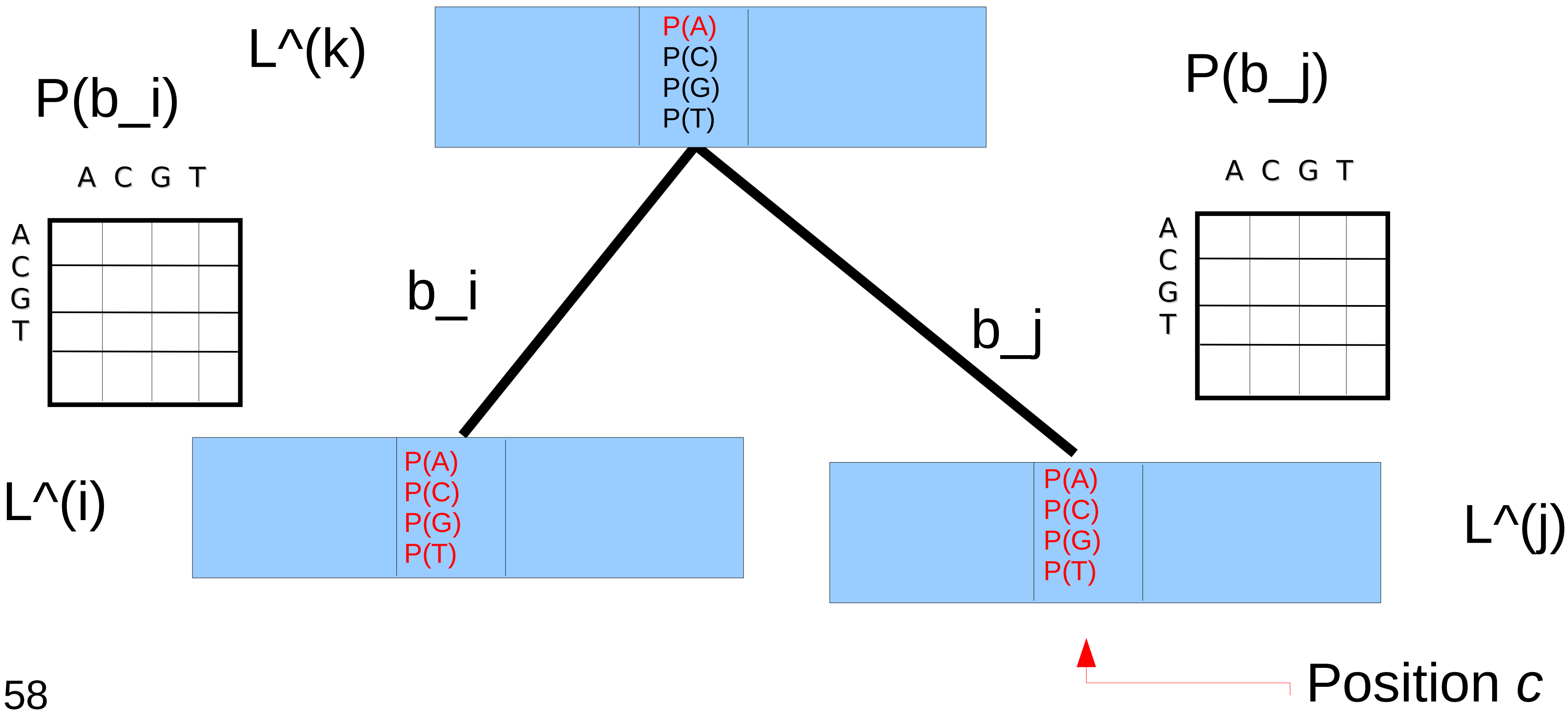
Felsenstein Pruning



Felsenstein Pruning

AND (left branch/right branch)

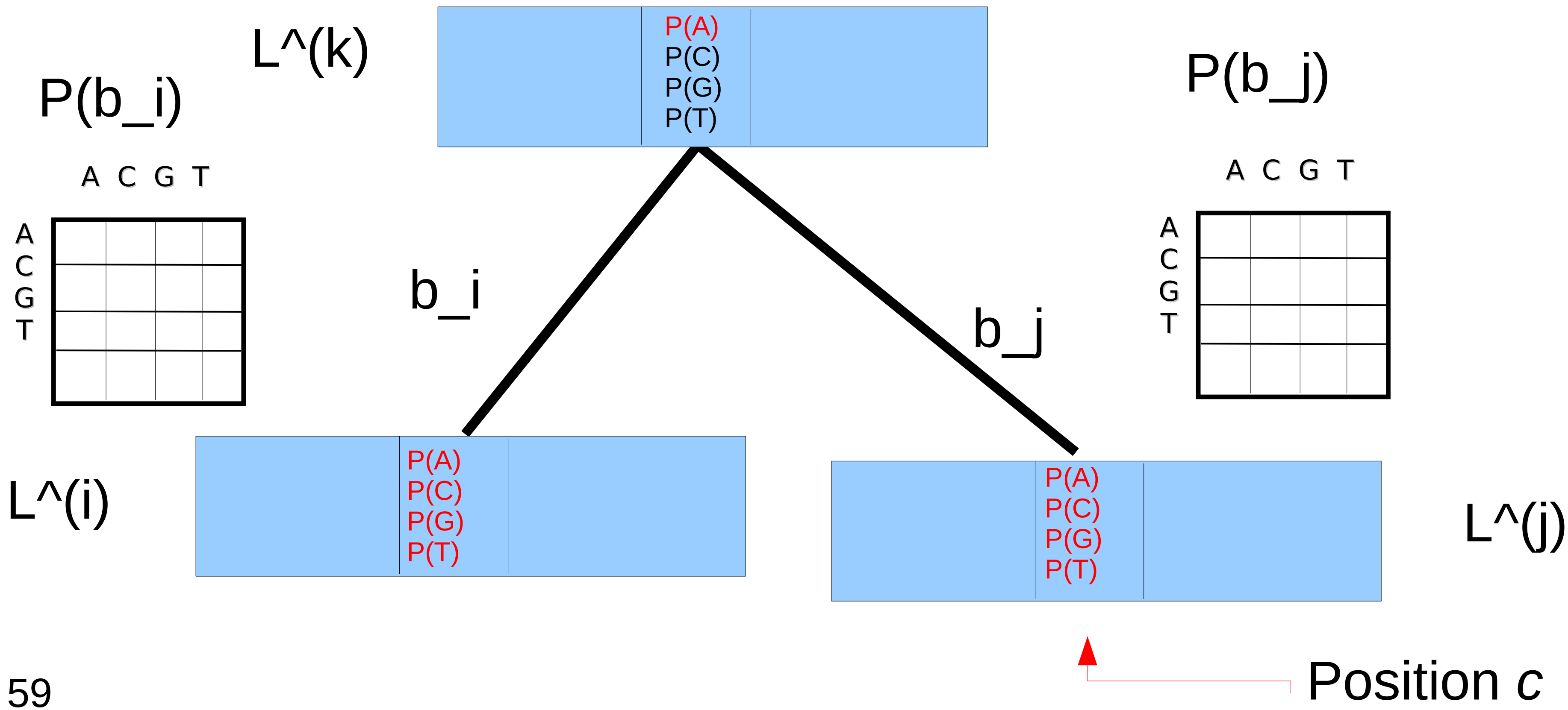
$$\vec{L}_A^{(k)}(c) = \Big(\sum_{S=A}^{T} P_{AS}(b_i)\vec{L}_S^{(i)}(c)\Big)\Big(\sum_{S=A}^{T} P_{AS}(b_j)\vec{L}_S^{(j)}(c)\Big)$$



Felsenstein Pruning

OR (along left branch)

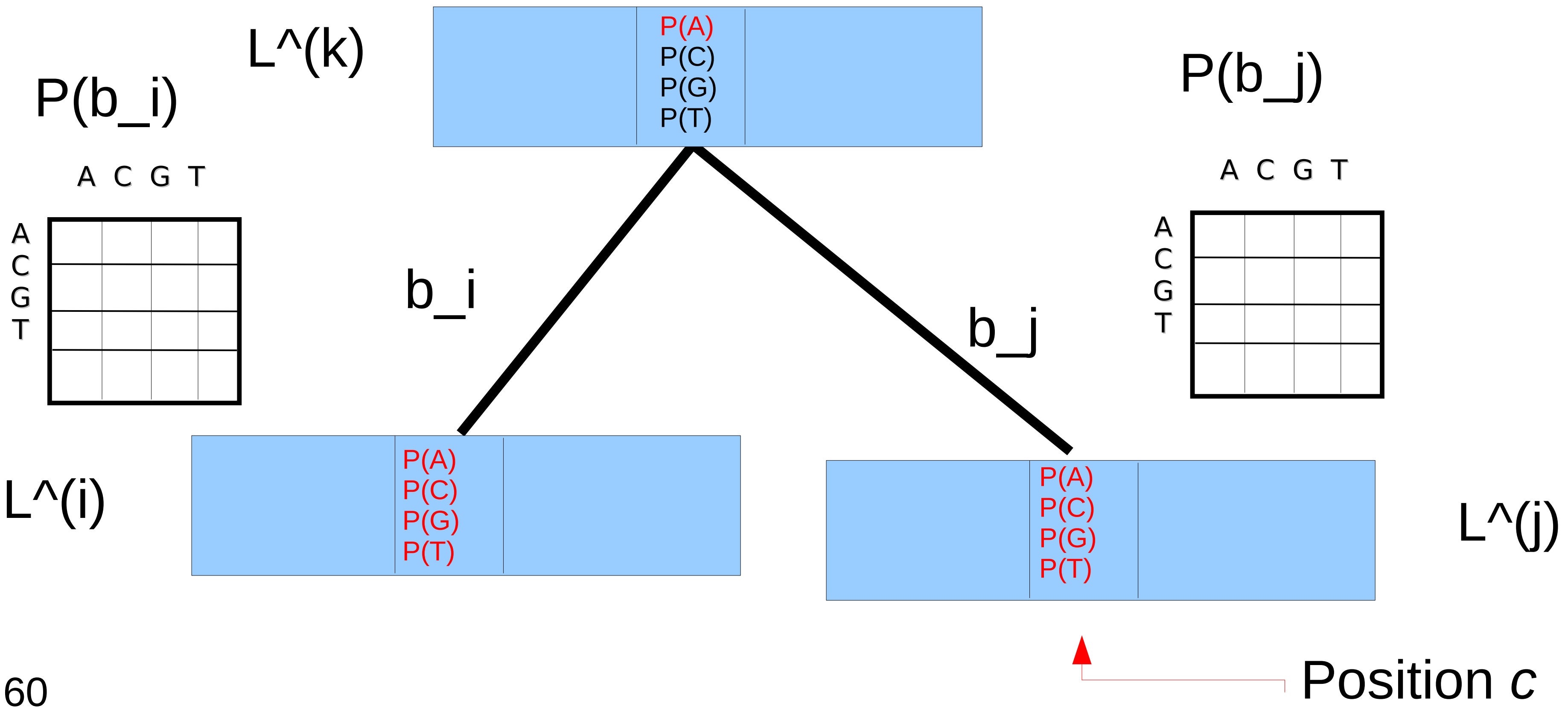
$$\vec{L}_A^{(k)}(c) = \left(\sum_{S=A}^{T} P_{AS}(b_i)\vec{L}_S^{(i)}(c)\right)\left(\sum_{S=A}^{T} P_{AS}(b_j)\vec{L}_S^{(j)}(c)\right)$$



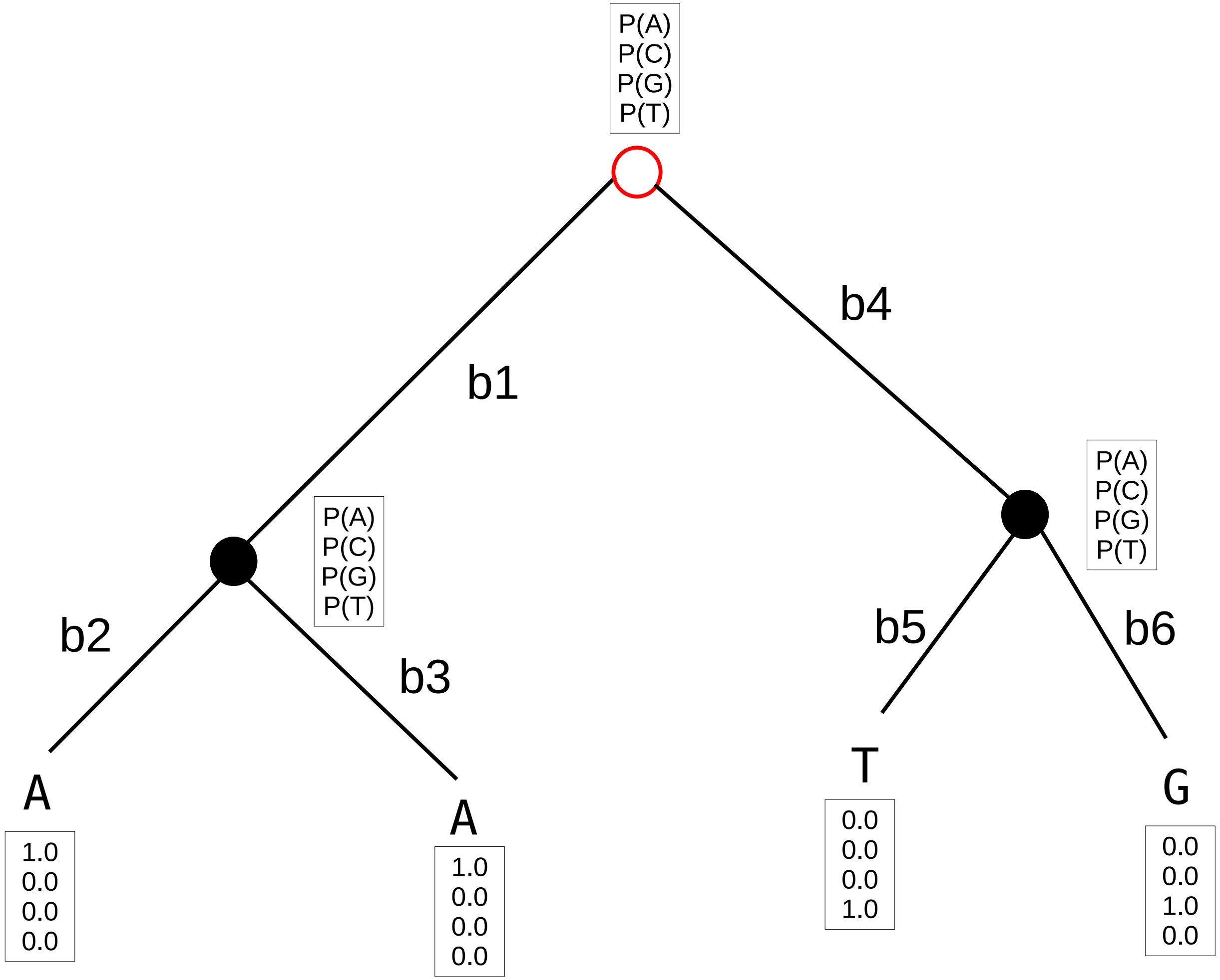
Felsenstein Pruning

OR (along right branch)

$$\vec{L}_A^{(k)}(c) = \Big(\sum_{S=A}^{T} P_{AS}(b_i)\vec{L}_S^{(i)}(c)\Big)\Big(\sum_{S=A}^{T} P_{AS}(b_j)\vec{L}_S^{(j)}(c)\Big)$$

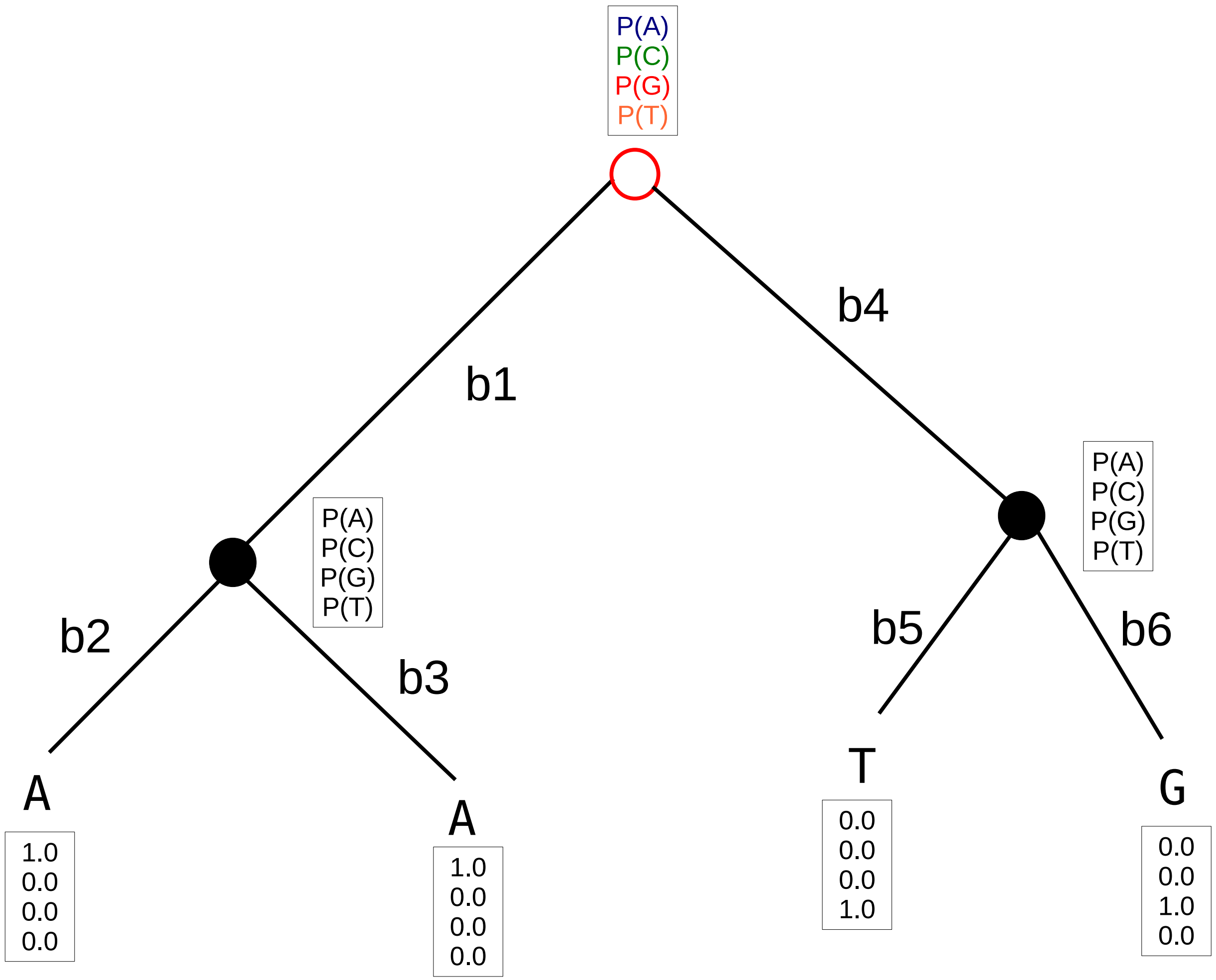


Felsenstein Pruning



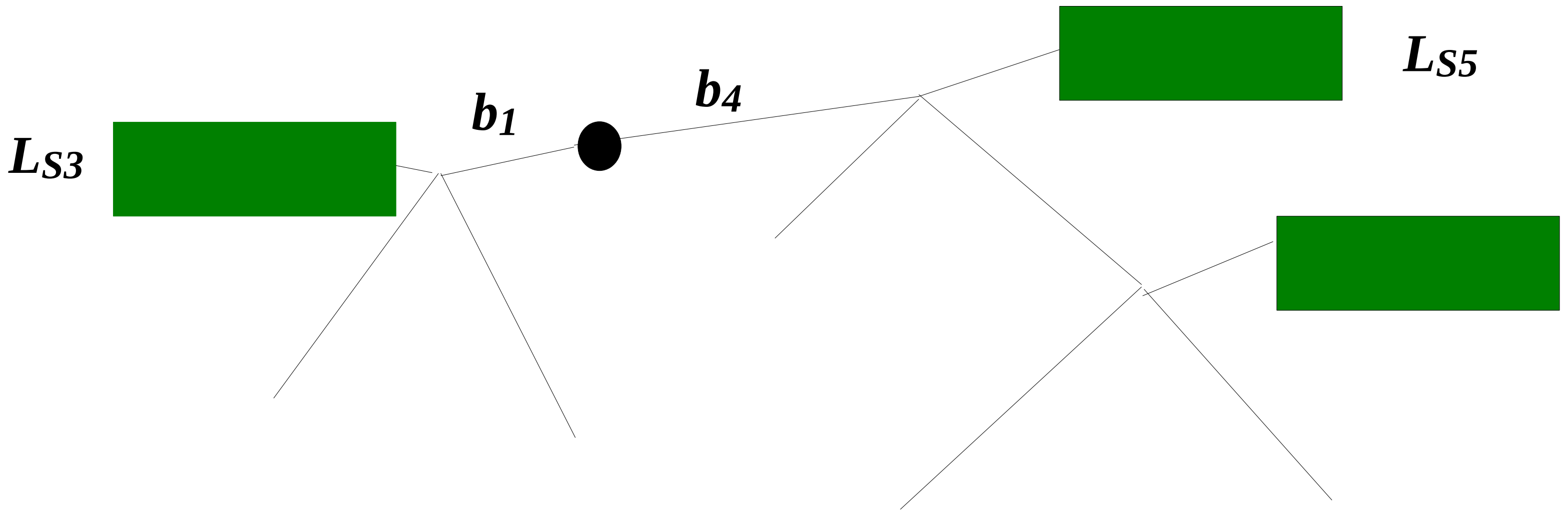
Felsenstein Pruning

Likelihood at the root: $L_i = \pi_A P(A) + \pi_C P(C) + \pi_G P(G) + \pi_T P(T)$



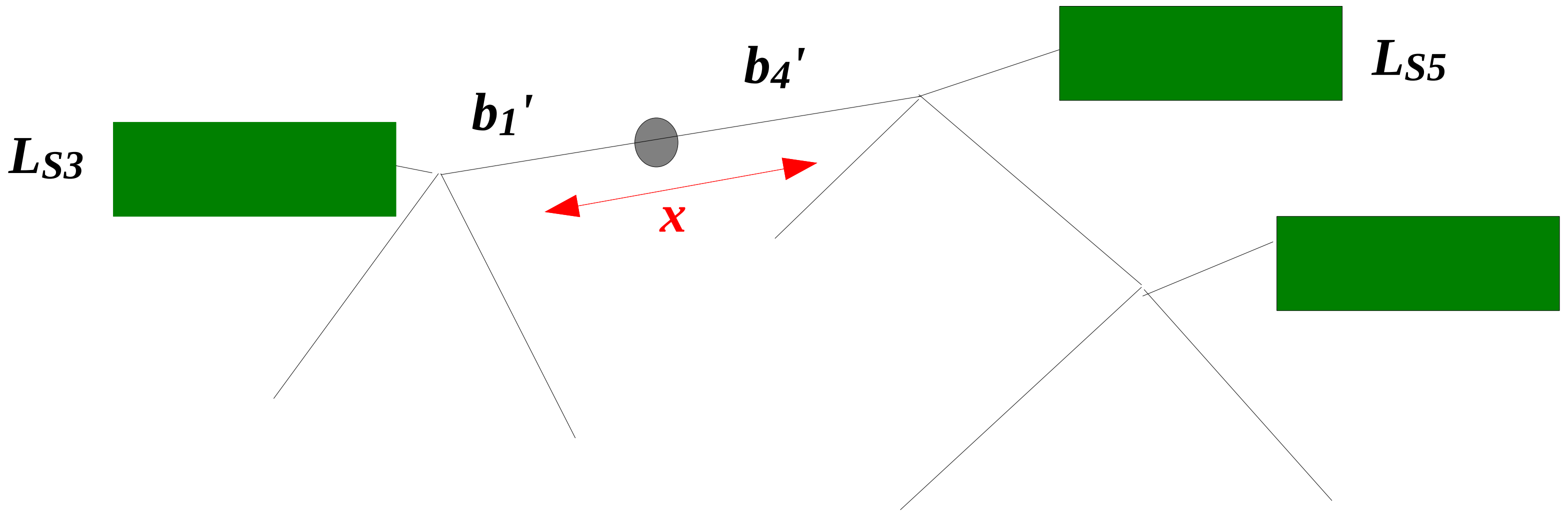
Why is time-reversibility important?

$$L = \sum_{S_4=A}^{T} \pi_{S_4} \sum_{S_3=A}^{T} P_{S_4 S_3}(b_1) L_{S_3}^{(3)} \sum_{S_5=A}^{T} P_{S_4 S_5}(b_4) L_{S_5}^{(5)}$$



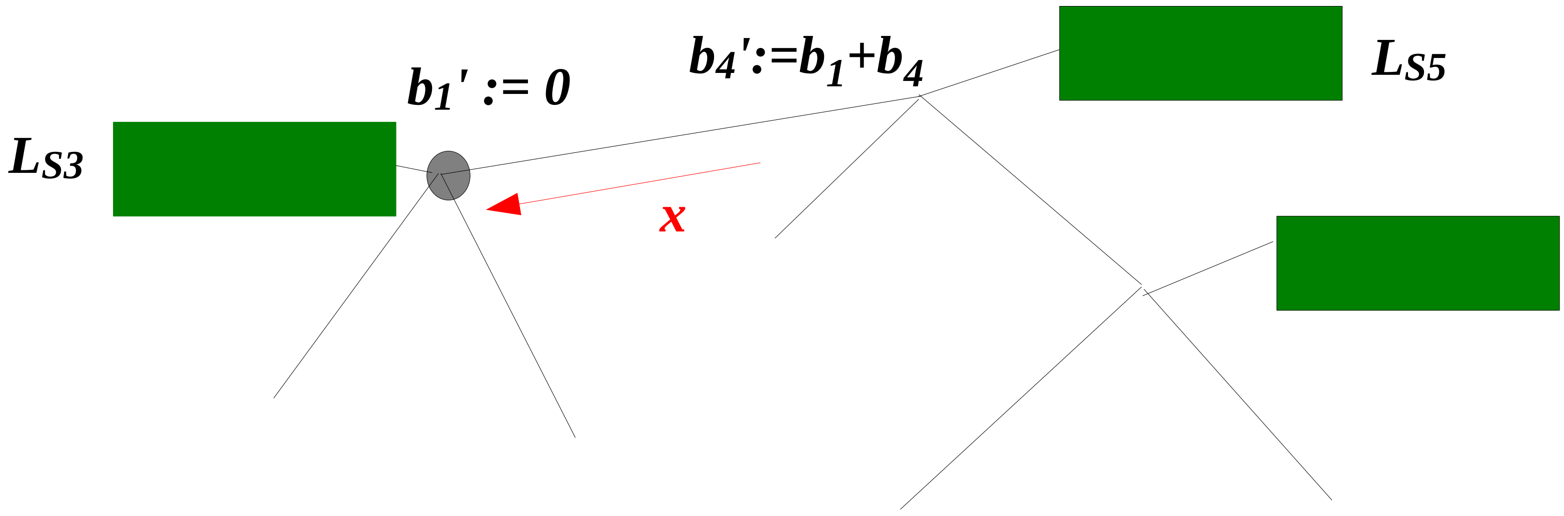
Why is time-reversibility important?

$$L = L' = \sum_{S_4=A}^{T} \pi_{S_4} \sum_{S_3=A}^{T} P_{S_4 S_3}(b_1 + x) L_{S_3}^{(3)} \sum_{S_5=A}^{T} P_{S_4 S_5}(b_4 - x) L_{S_5}^{(5)}$$



Why is time-reversibility important?

$$L = L' = \sum_{S_4=A}^{T} \pi_{S_4} \sum_{S_3=A}^{T} P_{S_4 S_3}(b_1 + x) L_{S_3}^{(3)} \sum_{S_5=A}^{T} P_{S_4 S_5}(b_4 - x) L_{S_5}^{(5)}$$



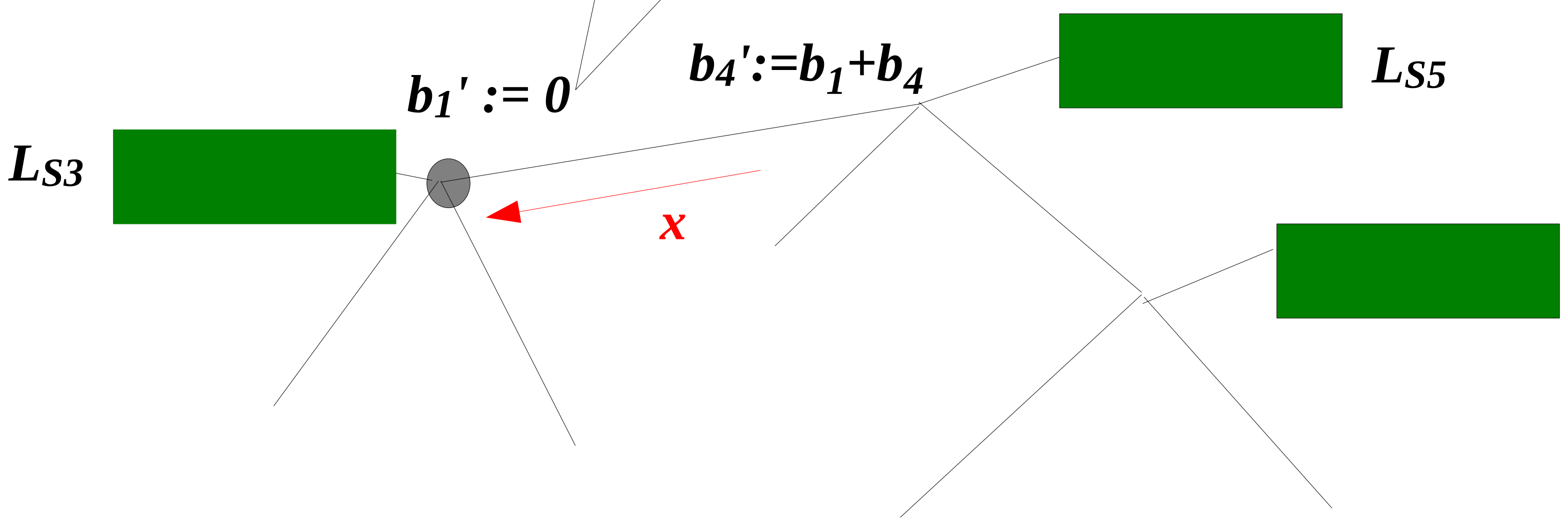
Why is time-reversibility important?

$$L = L' = \sum_{S_4=A}^{T} \cdots \sum_{S_5=A}^{T} P_{S_4 S_5}(b_4 - x) L^{(5)}_{S_5}$$

This observation can be applied recursively to the tree

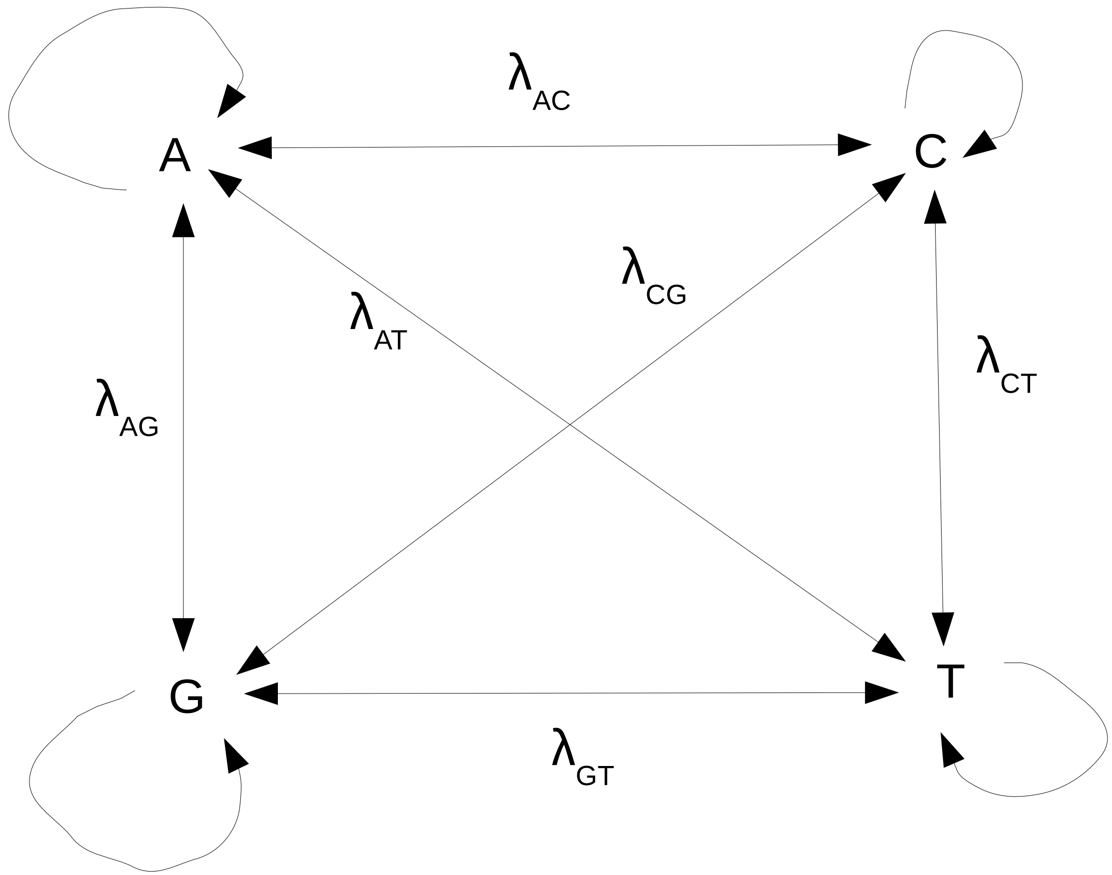
→

It does not matter at all where we place the root!



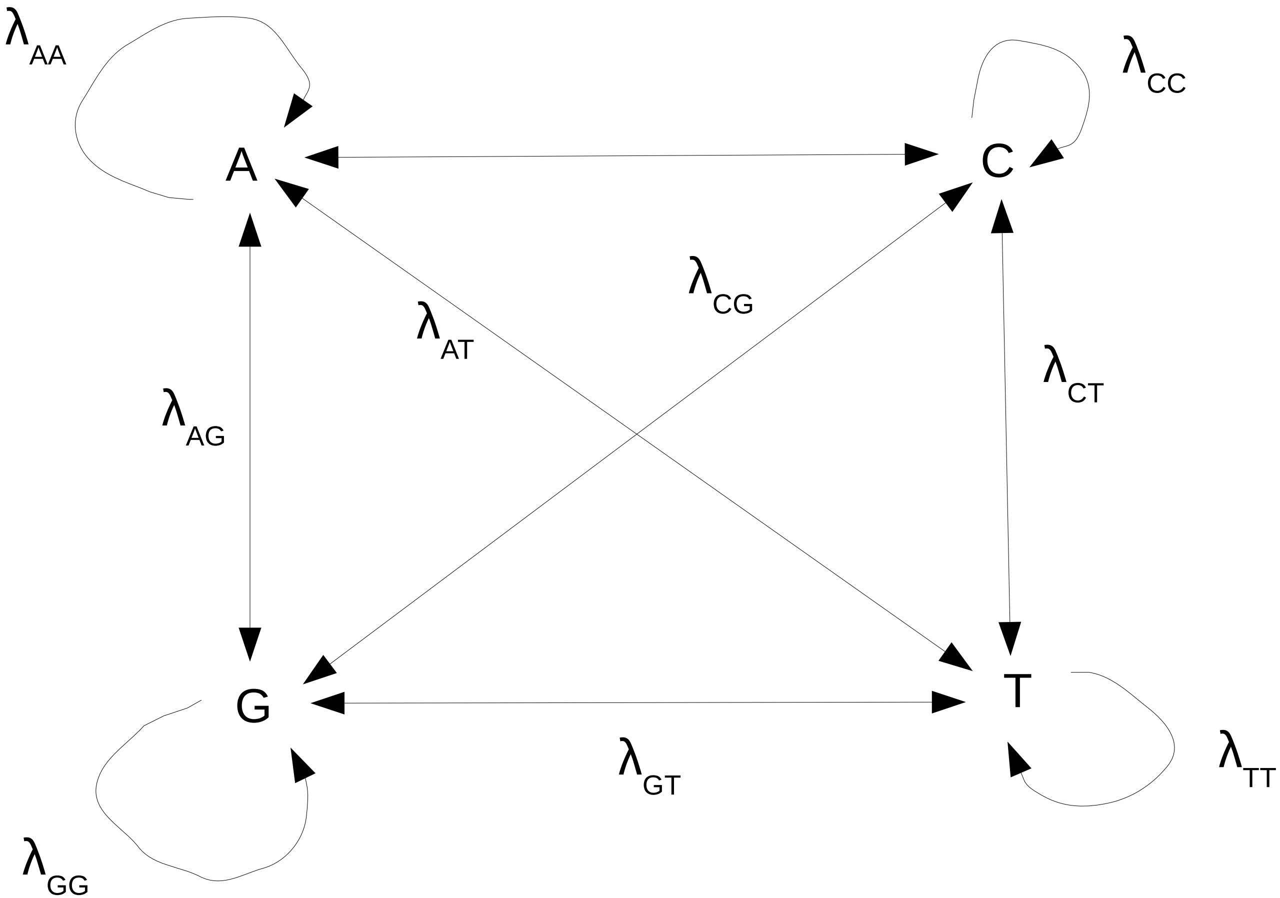
What's in the black box $P_{ij}(t)$?

Instantaneous rate matrix *R*!



What's in the black box $P_{ij}(t)$?

What about the probabilities of staying in the current state?
→ they are given by the properties of continuous Markov chains!
e.g., $\lambda_{AA} = -(\lambda_{AC} + \lambda_{AG} + \lambda_{AT})$ → remember from lecture on Markov models: rows in the *R* matrix need to sum to **0**



What's in the black box $P_{ij}(t)$?

	A	C	G	T
A	*	λ_{AC}	λ_{AG}	λ_{AT}
C		*	λ_{CG}	λ_{CT}
G			*	λ_{GT}
T	Symmetric			*

What's in the black box $P_{ij}(t)$?

Diagonal values are
given by the off-diagonal
values (R matrix property)
$\lambda_{AA} = -(\lambda_{AC} + \lambda_{AG} + \lambda_{AT})$

	A	C	G	T
A	*	λ_{AC}	λ_{AG}	λ_{AT}
C		*	λ_{CG}	λ_{CT}
G			*	λ_{GT}
T	Symmetric			*

What's in the black box $P_{ij}(t)$?

$$
\begin{array}{c} \\ A \\ C \\ G \\ T \end{array}
\begin{array}{cccc} A & C & G & T \\ \end{array}
$$

$$
\begin{array}{c|cccc}
 & A & C & G & T \\
\hline
A & * & \lambda_{AC} & \lambda_{AG} & \lambda_{AT} \\
C & & * & \lambda_{CG} & \lambda_{CT} \\
G & & & * & \lambda_{GT} \\
T & & \text{Symmetric} & & *
\end{array}
$$

Equilibrium frequency vector $\pi = (\pi_A,\ \pi_C,\ \pi_G,\ \pi_T)$ *where* $\pi_A + \pi_C + \pi_G + \pi_T = 1$

The simple Jukes-Cantor model

	A	C	G	T
A	*	λ	λ	λ
C		*	λ	λ
G			*	λ
T				*

Π = (1/4, 1/4, 1/4, 1/4)

The Felsenstein 81 model

	A	C	G	T
A	*	λ	λ	λ
C		*	λ	λ
G			*	λ
T				*

$\Pi_i \neq \Pi_j$

Kimura 2-parameter model 1980

$$
\begin{array}{c|cccc}
 & A & C & G & T \\
\hline
A & * & \lambda & \zeta & \lambda \\
C & & * & \zeta & \lambda \\
G & & & * & \zeta \\
T & & & & *
\end{array}
$$

$\Pi = (1/4,\ 1/4,\ 1/4,\ 1/4)$

HKY85

	A	C	G	T
A	*	λ	ζ	λ
C		*	ζ	λ
G			*	ζ
T				*

$\Pi_i \neq \Pi_j$

GTR 1986

$$
\begin{array}{c|cccc|}
 & A & C & G & T \\
\hline
A & * & \alpha & \beta & \gamma \\
C & & * & \delta & \varepsilon \\
G & & & * & \zeta \\
T & & & & *
\end{array}
$$

$\Pi_i \neq \Pi_j$

GTR 1986

	A	C	G	T
A	*	α	β	γ
C		*	δ	ε
G			*	ζ
T				*

Note that these are **relative** rates, their Values only matter relative to each other, so we can set *ζ := 1.0* by default

$\Pi_i \neq \Pi_j$

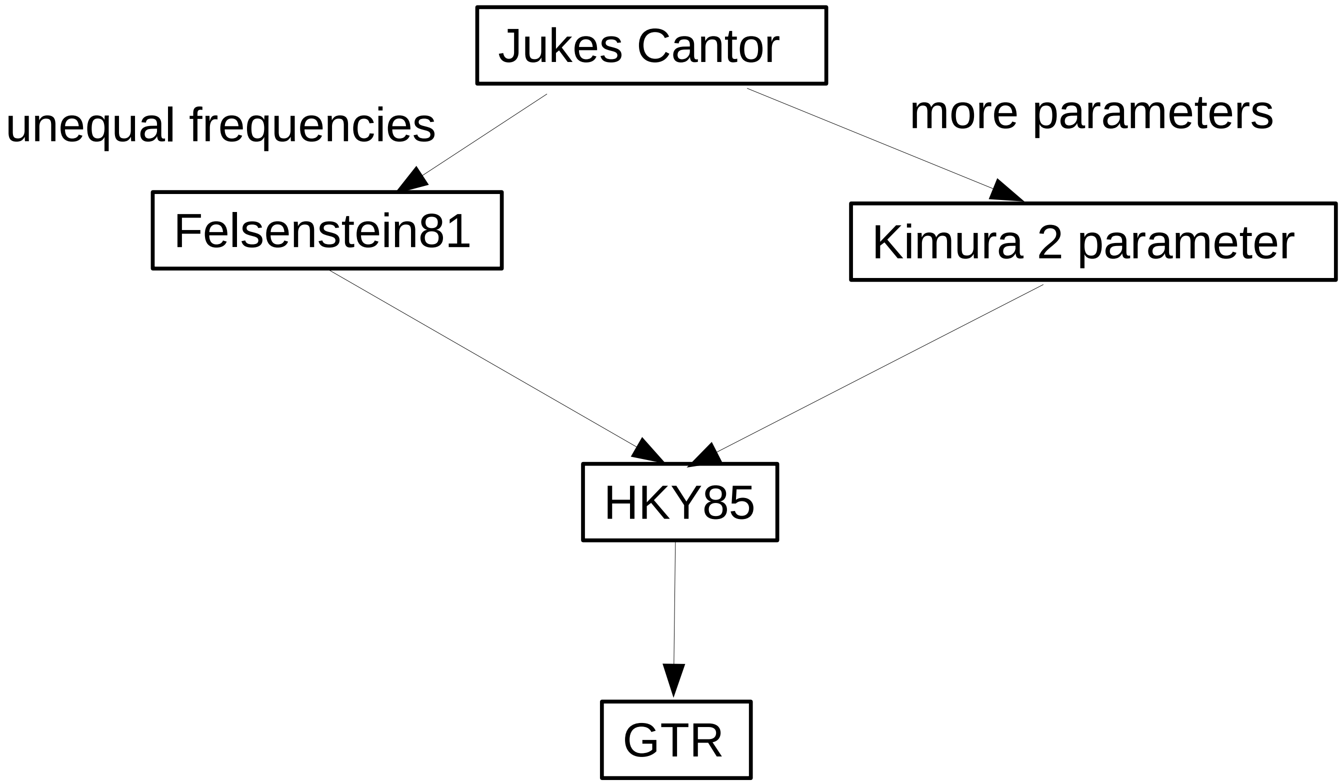
GTR 1986

	A	C	G	T
A	*	α	β	γ
C		*	δ	ε
G			*	1.0
T				*

Note that these are **relative** rates, their values only matter relative to each other, so we can set $\zeta := 1.0$ by default. Although the GTR model has 6 rates, it only has 5 free parameters!

$\Pi_i \neq \Pi_j$

Model Hierarchy



GTR 1986

This is a rate matrix, time reversibility would require $\boldsymbol{\pi_i r_{ij} = \pi_j r_{ji}}$

	A	C	G	T
A	*	α	β	γ
C		*	δ	ε
G			*	1.0
T				*

$\Pi_i \neq \Pi_j$

GTR 1986

	A	C	G	T
A	*	α	β	γ
C		*	δ	ε
G			*	1.0
T				*

This is a rate matrix, time reversibility would require $\boldsymbol{\pi_i r_{ij} = \pi_j r_{ji}}$

Solution: introduce a *Q* matrix *Q := diag(π) R*

$$\begin{pmatrix} \pi_A & & & \\ & \pi_C & & \\ & & \pi_G & \\ & & & \pi_T \end{pmatrix}$$

$\Pi_i \neq \Pi_j$

GTR 1986

This is a rate matrix, time reversibility would require $\boldsymbol{\pi_i r_{ij} = \pi_j r_{ji}}$

Solution: introduce a *Q* matrix *Q := diag(π) R*

	A	C	G	T
A	*	α	β	γ
C		*	δ	ε
G			*	1.0
T				*

$$\begin{pmatrix} \pi_A & & & \\ & \pi_C & & \\ & & \pi_G & \\ & & & \pi_T \end{pmatrix}$$

$\Pi_i \neq \Pi_j$

Then, $\pi_i r_{ij} = \pi_j r_{ji}$ holds

So how do we compute P(t) from Q?

- As we have seen in the lecture on Markov chains:

  $P(t) = e^{Qt} = I + Qt + 1/2!\,(Qt)^2 + 1/3!\,(Qt)^3 + \ldots$

- but this is unfortunately a matrix exponential :-(
- I will spare you the details, but in general, e.g., for GTR we need to apply an Eigenvector/Eigenvalue decomposition of Q to calculate:

  $P(t) = U \exp(diag(\lambda_i)t)\, U^{-1}$

Matrix and inverse matrix of eigenvectors of *Q*

So how do we compute P(t) from Q?

- As we have seen in the lecture on Markov chains:

  $P(t) = e^{Qt} = I + Qt + 1/2!\,(Qt)^2 + 1/3!\,(Qt)^3 + \dots$

- but this is unfortunately a matrix exponential :-(
- I will spare you the details, but in general, e.g., for GTR we need to apply an Eigenvector/Eigenvalue decomposition of Q to calculate:

  $P(t) = U \exp(\mathrm{diag}(\lambda_i)t)\, U^{-1}$

Diagonal matrix of eigenvalues of *Q*, here the exponential function *exp()* is invoked on scalar values!

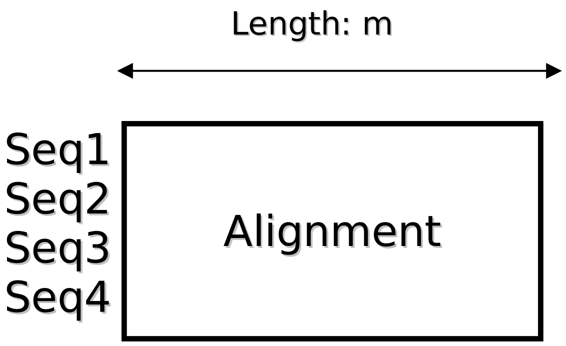
Likelihood Calculations

- So far, we have only seen how to calculate **a** likelihood on a
  - given, fixed tree topology
  - with given fixed branch lengths
  - and given, fixed remaining model parameters
- Computing the **maximum** likelihood score, is much more complicated as it requires
  1. functions for optimizing continuous parameters
  2. functions for searching the discrete space of trees

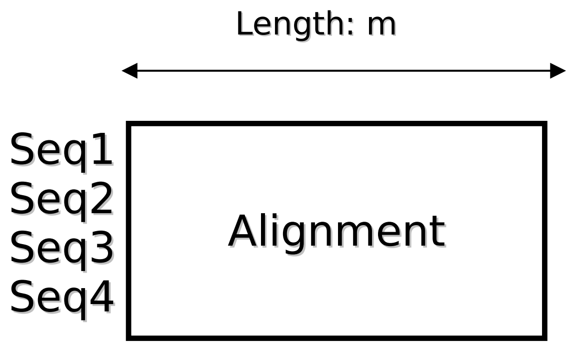
Outline – Lecture 10

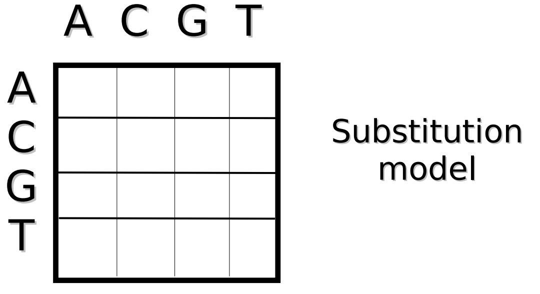
- Maximum Likelihood – motivation
- Computing the Likelihood on a tree
- **Computing the Maximum Likelihood on a tree**

Maximum Likelihood

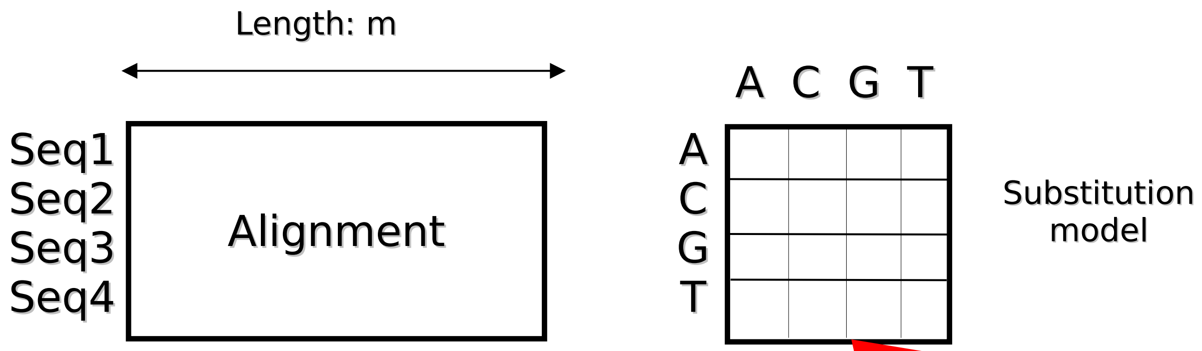


Maximum Likelihood



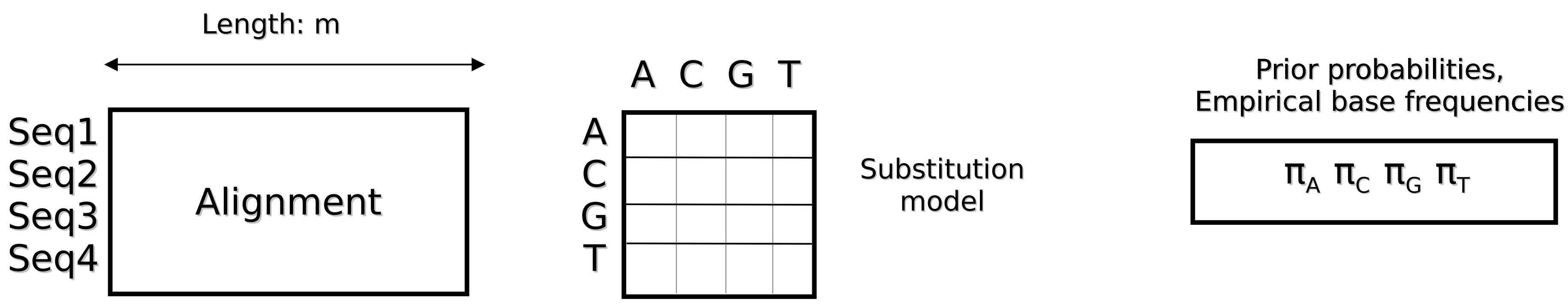


Maximum Likelihood

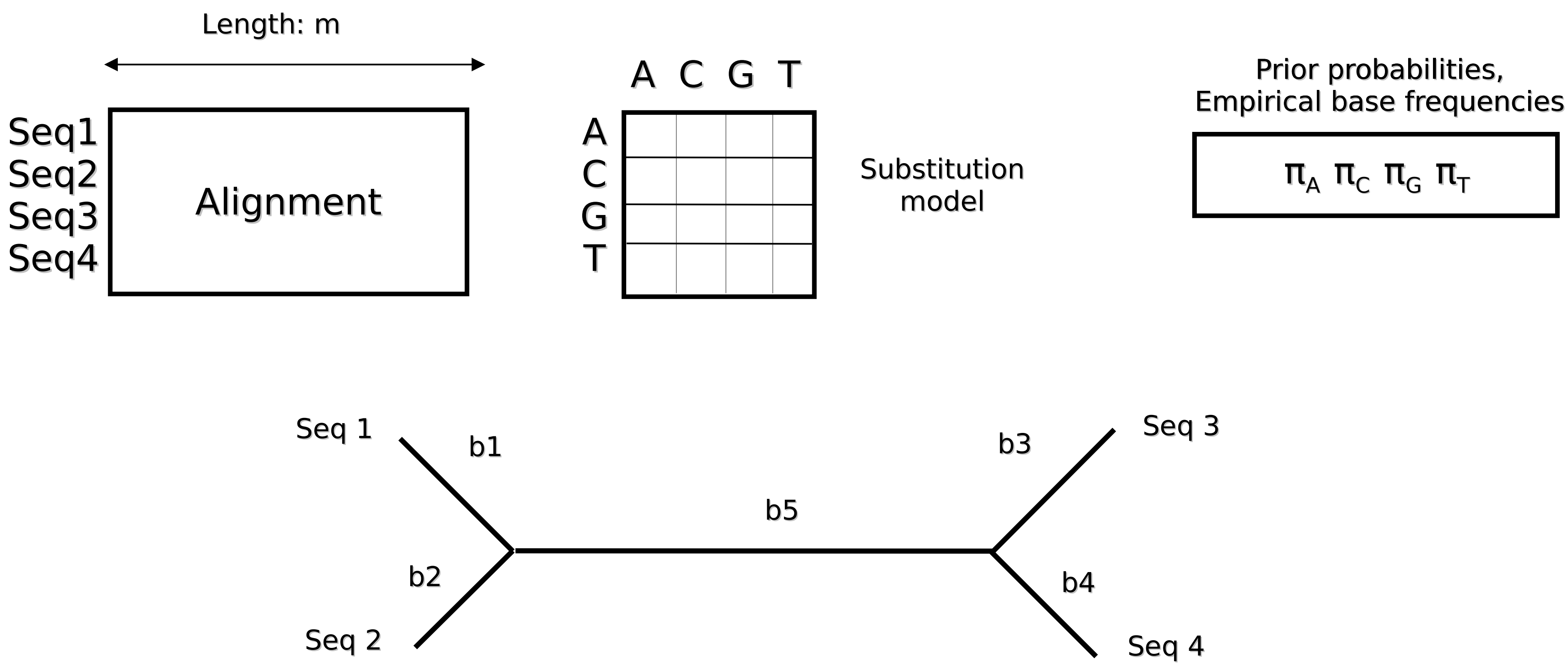


Commonly denoted as *Q* matrix: transition probs for time *dt*, for time *t*: $P(t)=e^{Qt}$

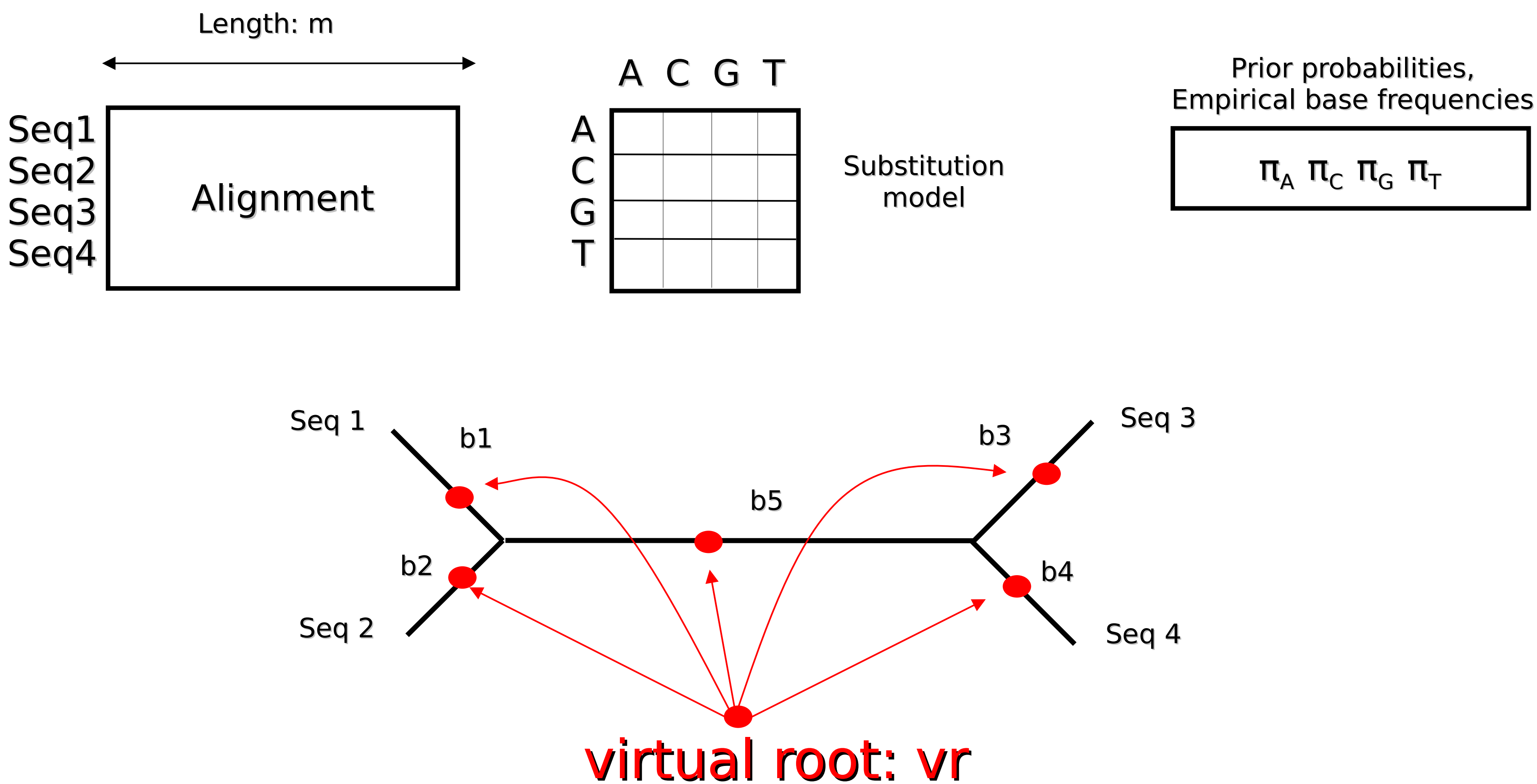
Maximum Likelihood



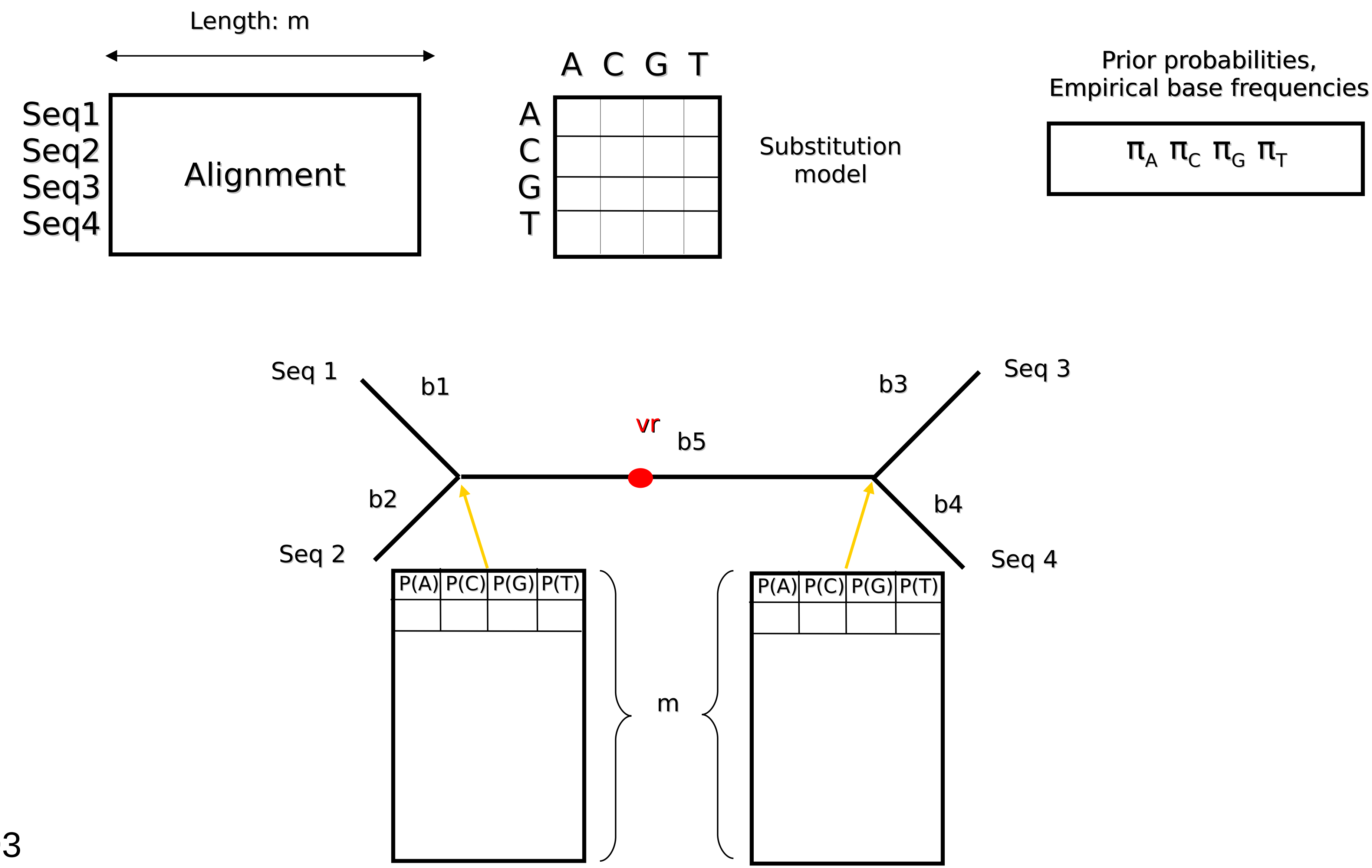
Maximum Likelihood



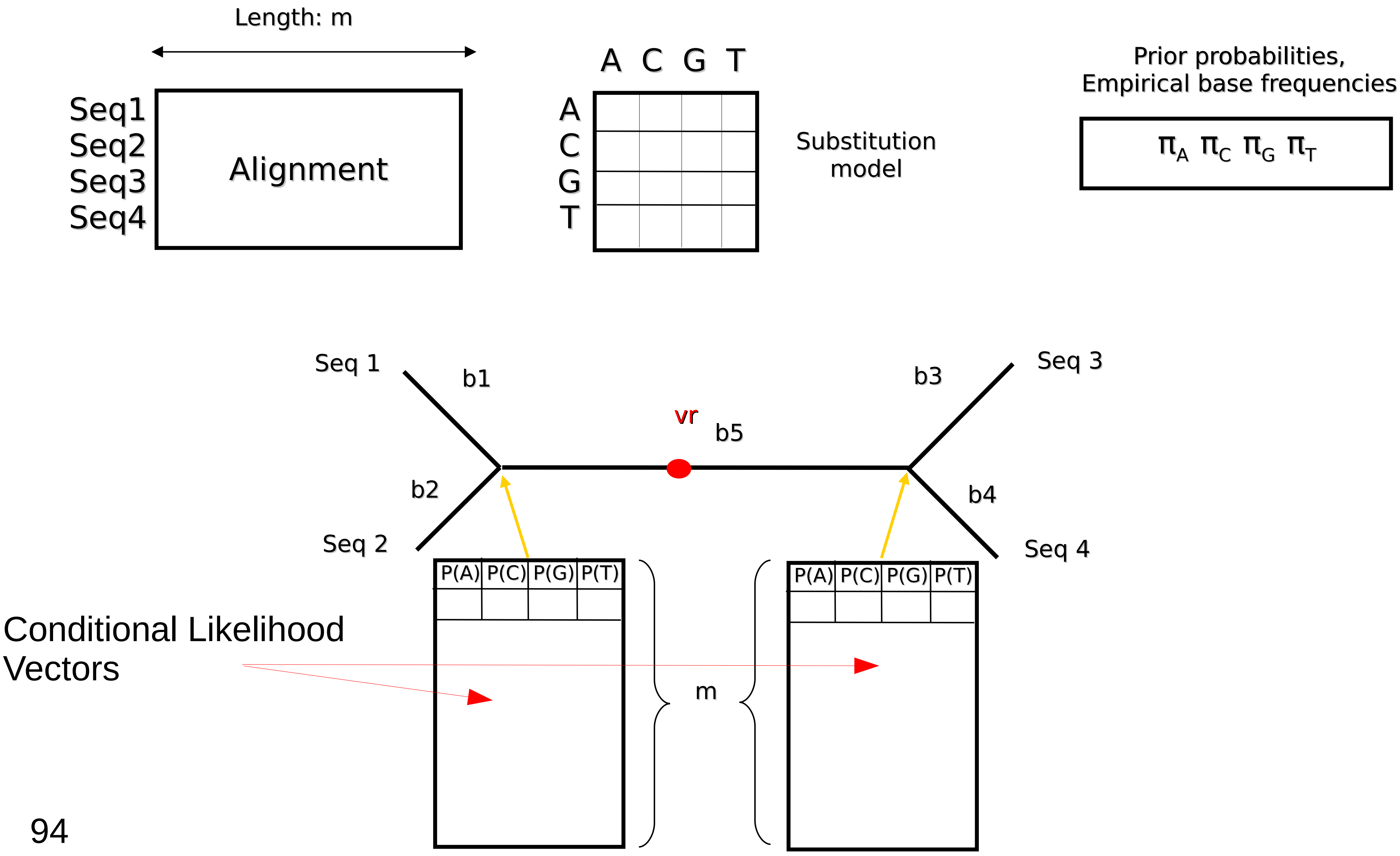
Maximum Likelihood



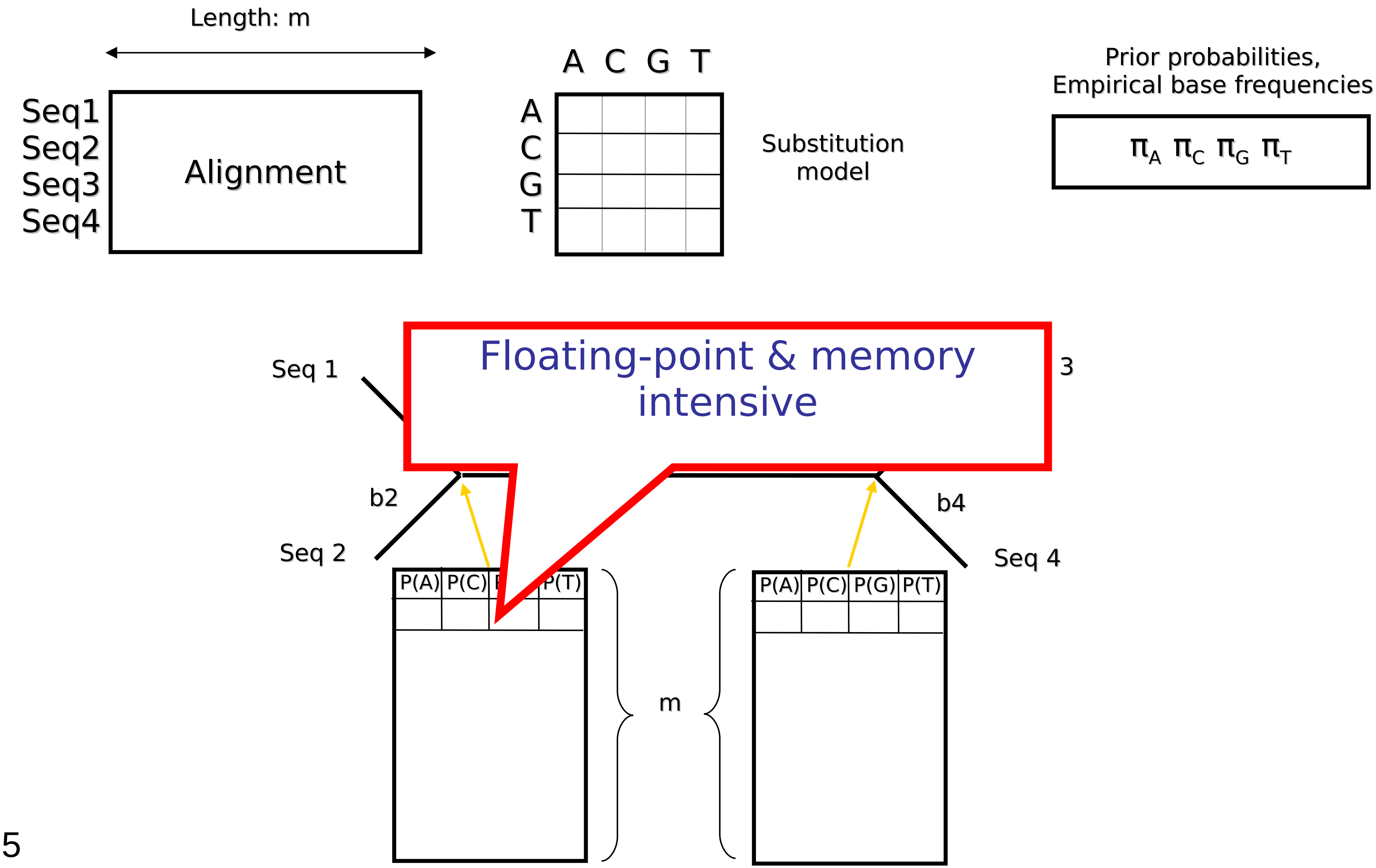
Maximum Likelihood



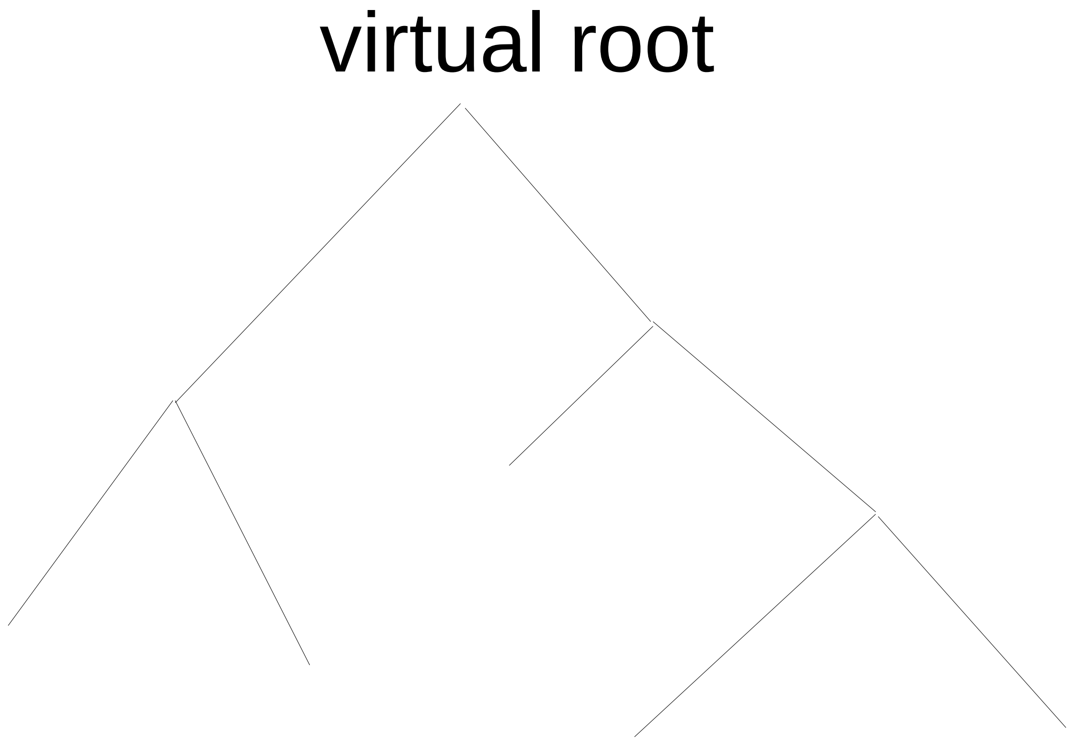
Maximum Likelihood



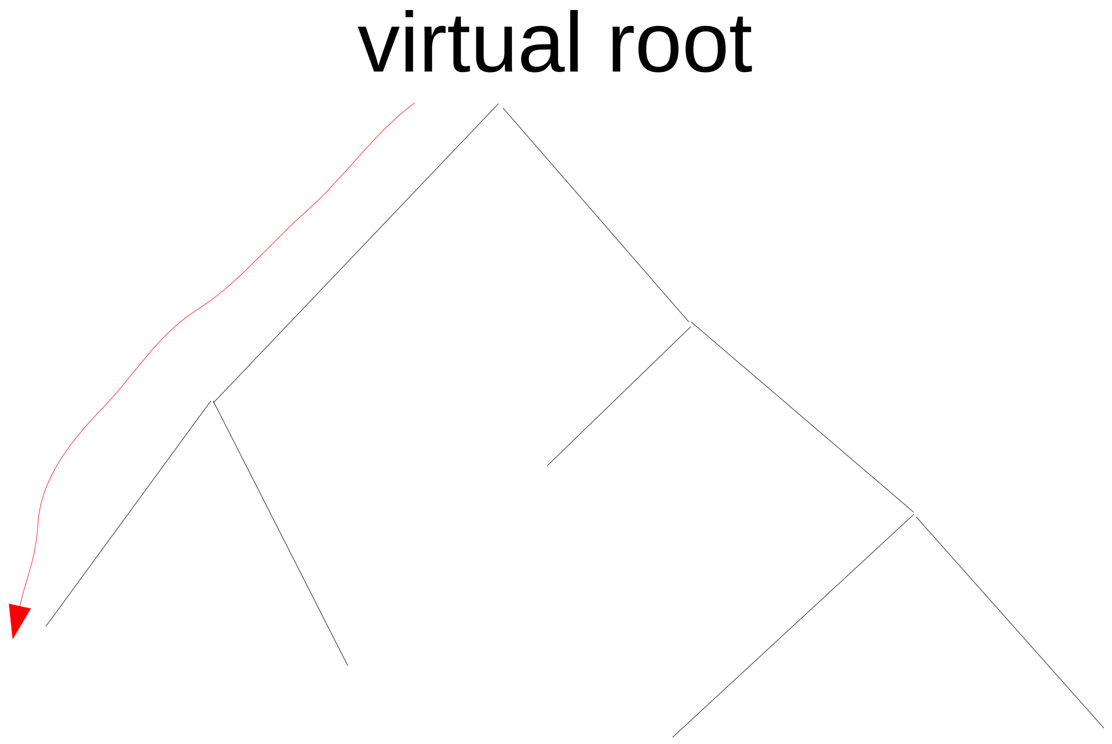
Maximum Likelihood



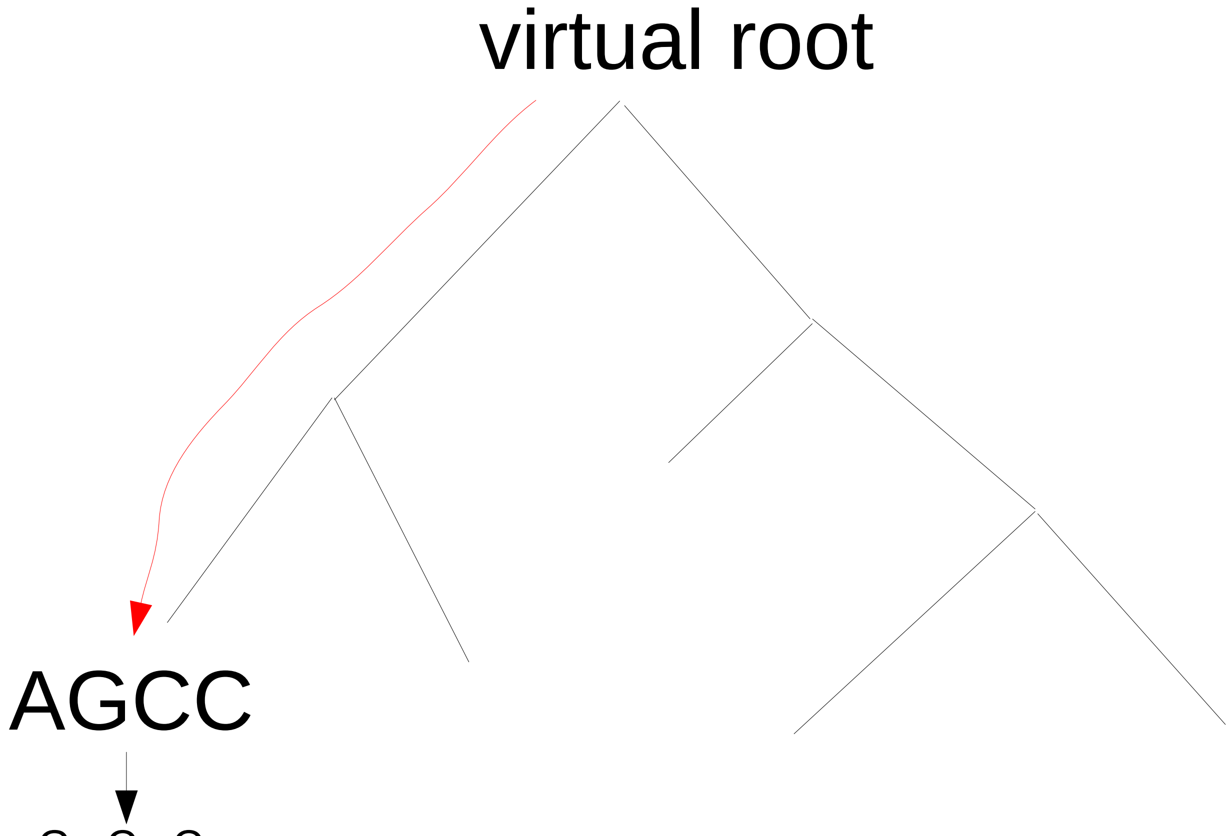
Post-order Traversal



Post-order Traversal



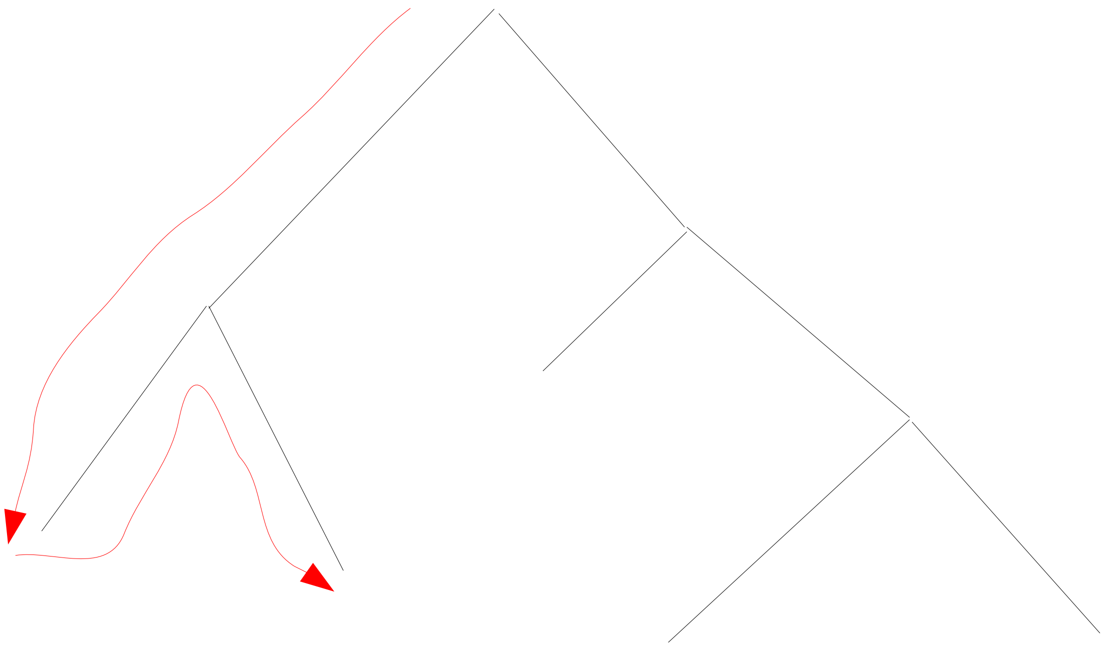
Post-order Traversal



A	G	C	C
1.0	0.0	0.0	0.0
0.0	0.0	**1.0**	**1.0**
0.0	**1.0**	0.0	0.0
0.0	0.0	0.0	0.0

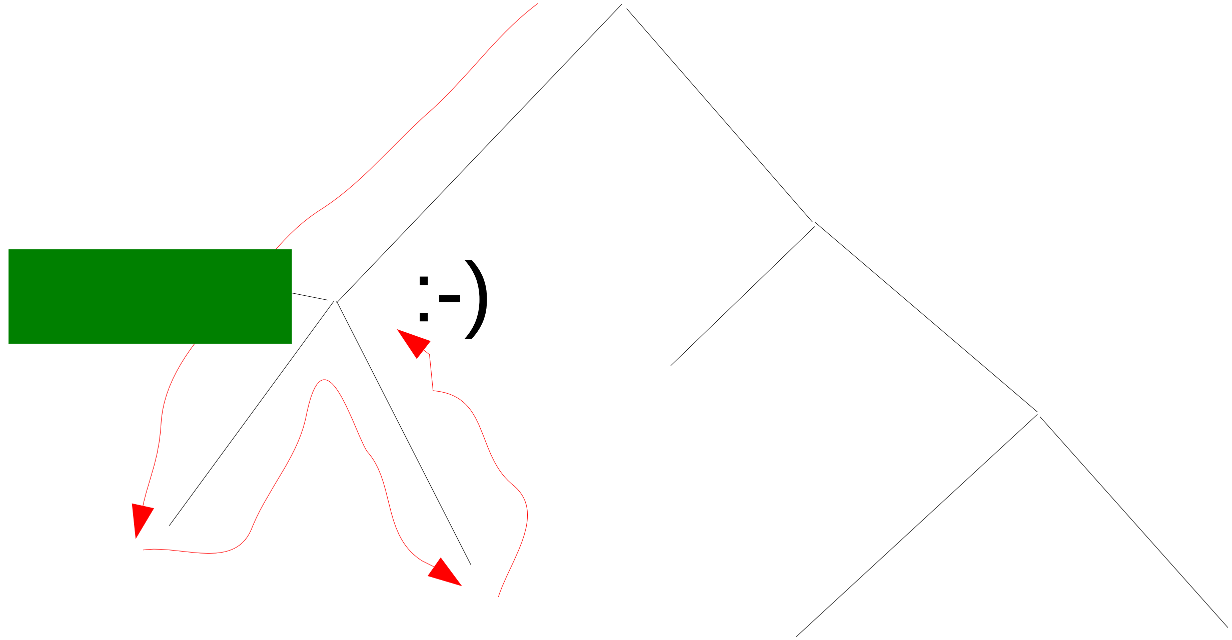
Post-order Traversal

virtual root



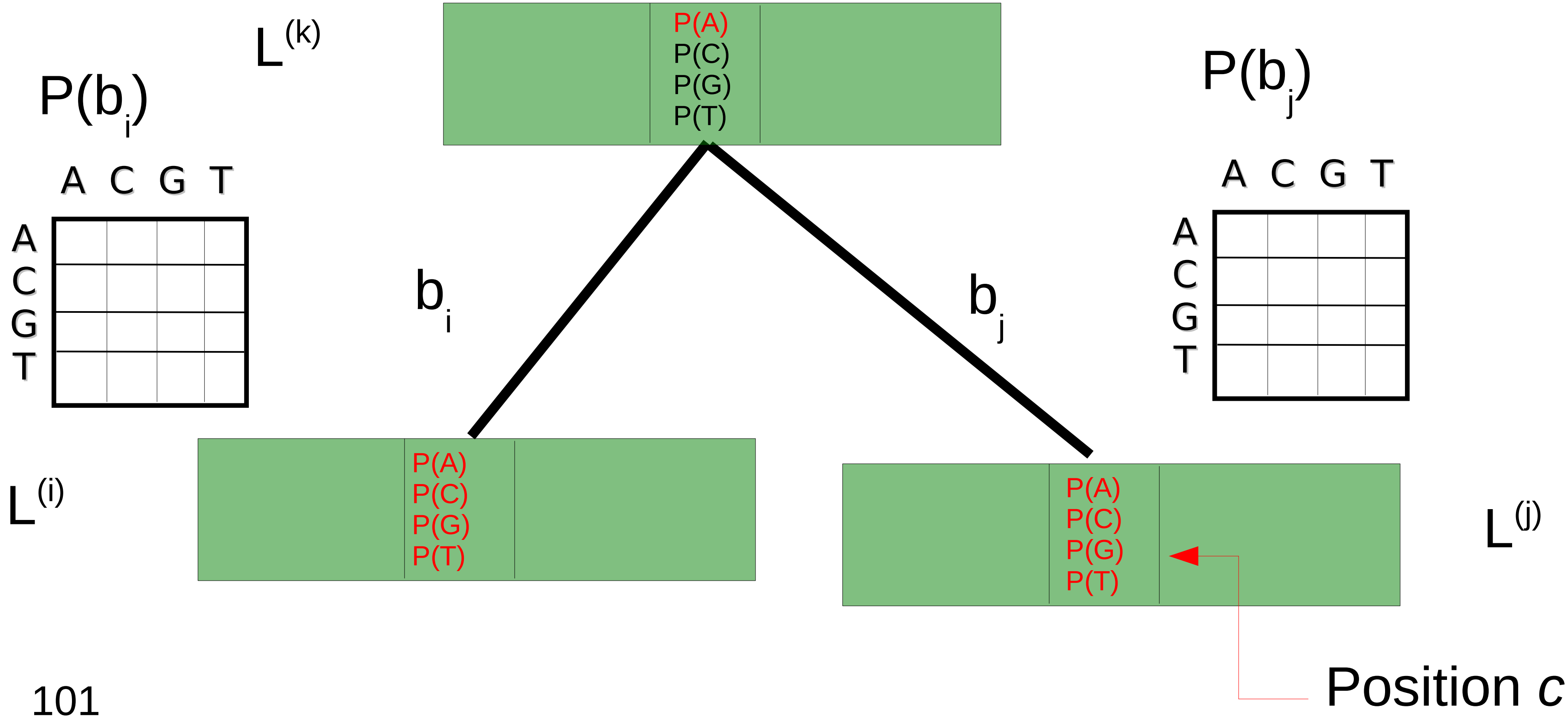
Post-order Traversal

virtual root

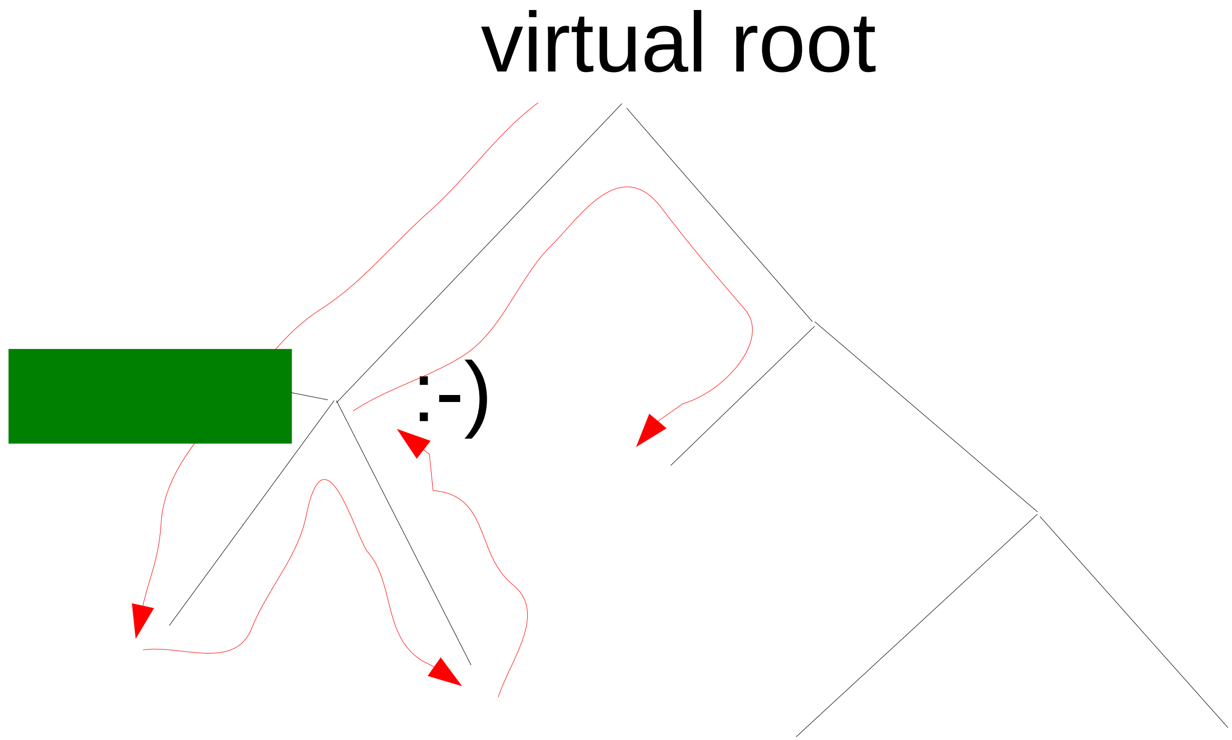


What happens when we compute this inner vector?

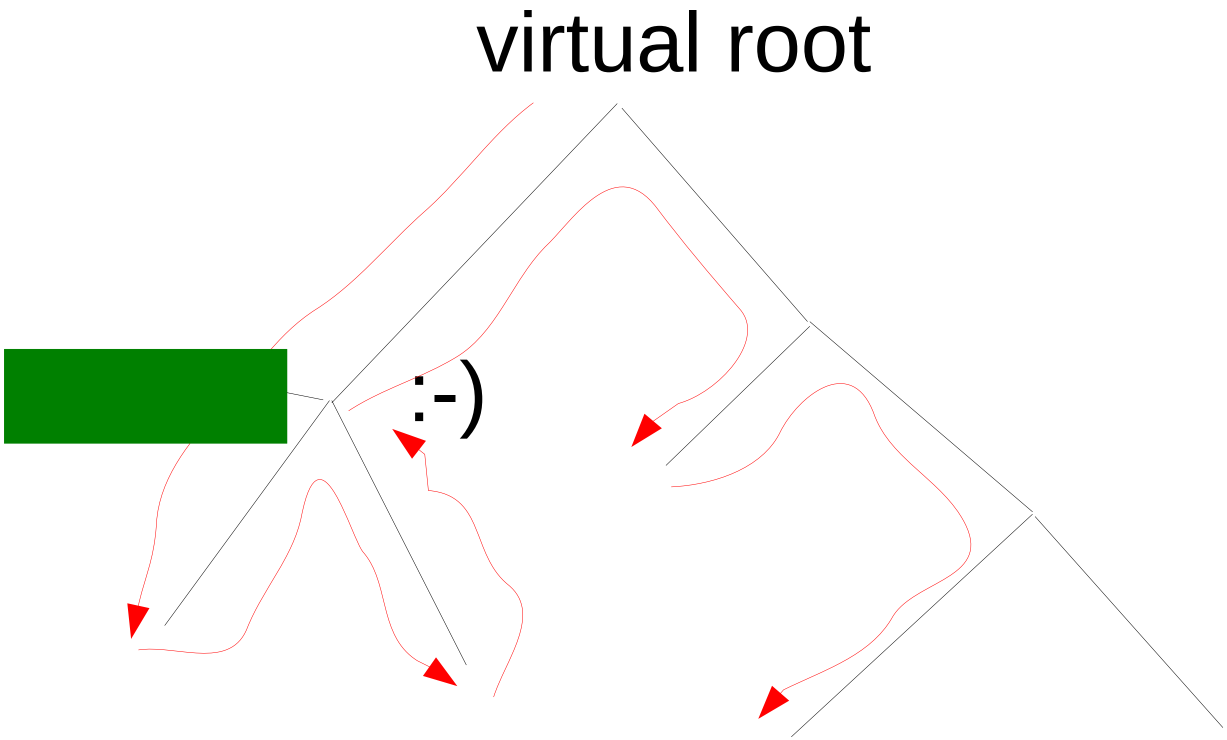
$$\vec{L}_A^{(k)}(c) = \Big(\sum_{S=A}^{T} P_{AS}(b_i)\vec{L}_S^{(i)}(c)\Big)\Big(\sum_{S=A}^{T} P_{AS}(b_j)\vec{L}_S^{(j)}(c)\Big)$$



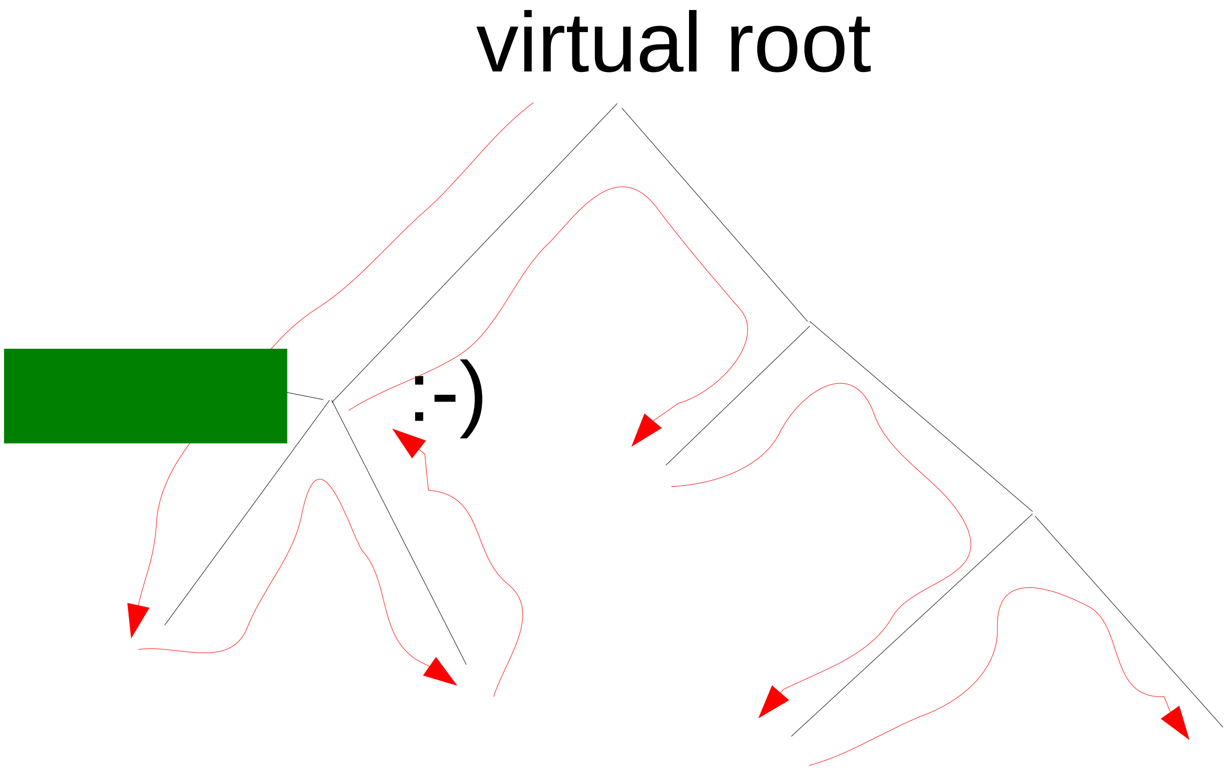
Post-order Traversal



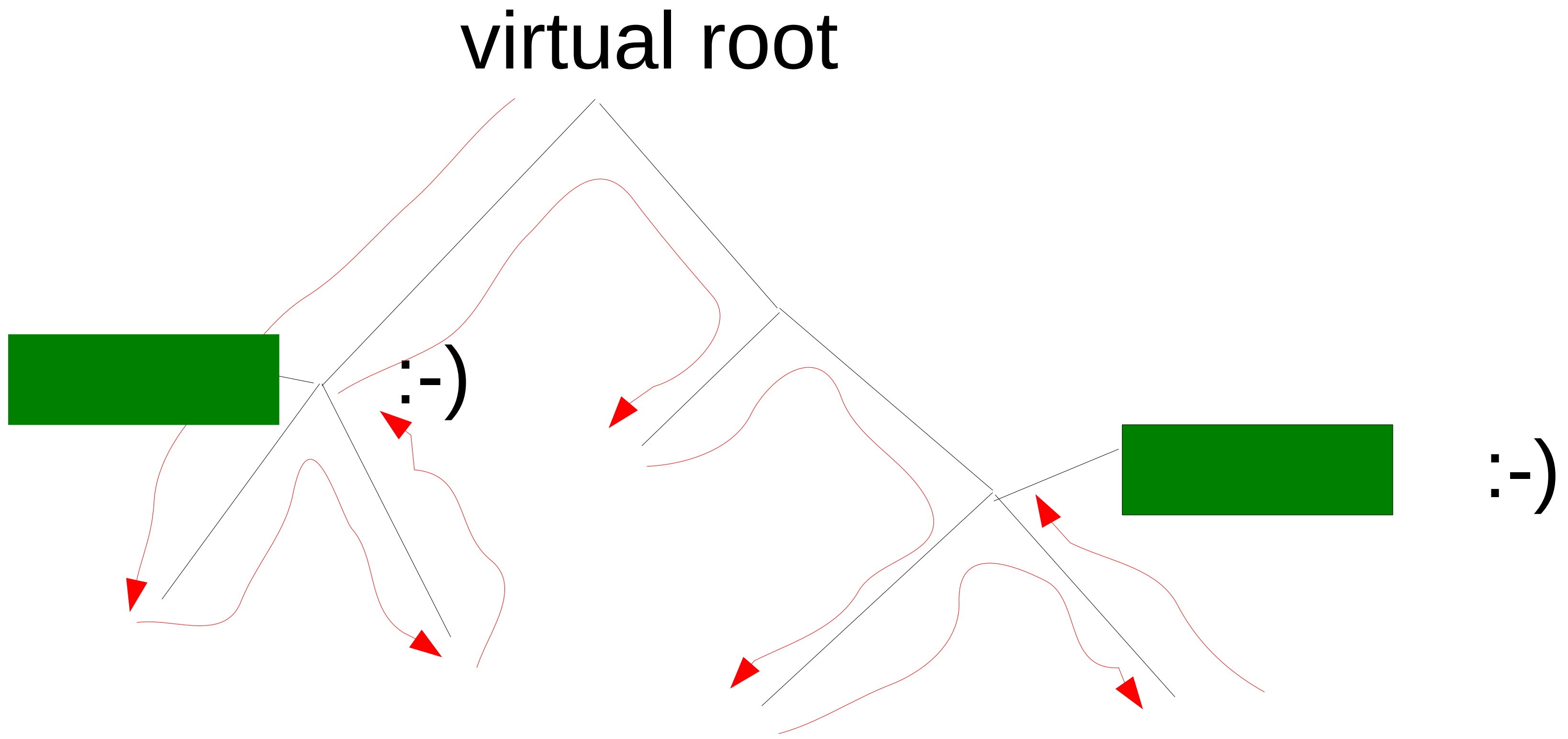
Post-order Traversal



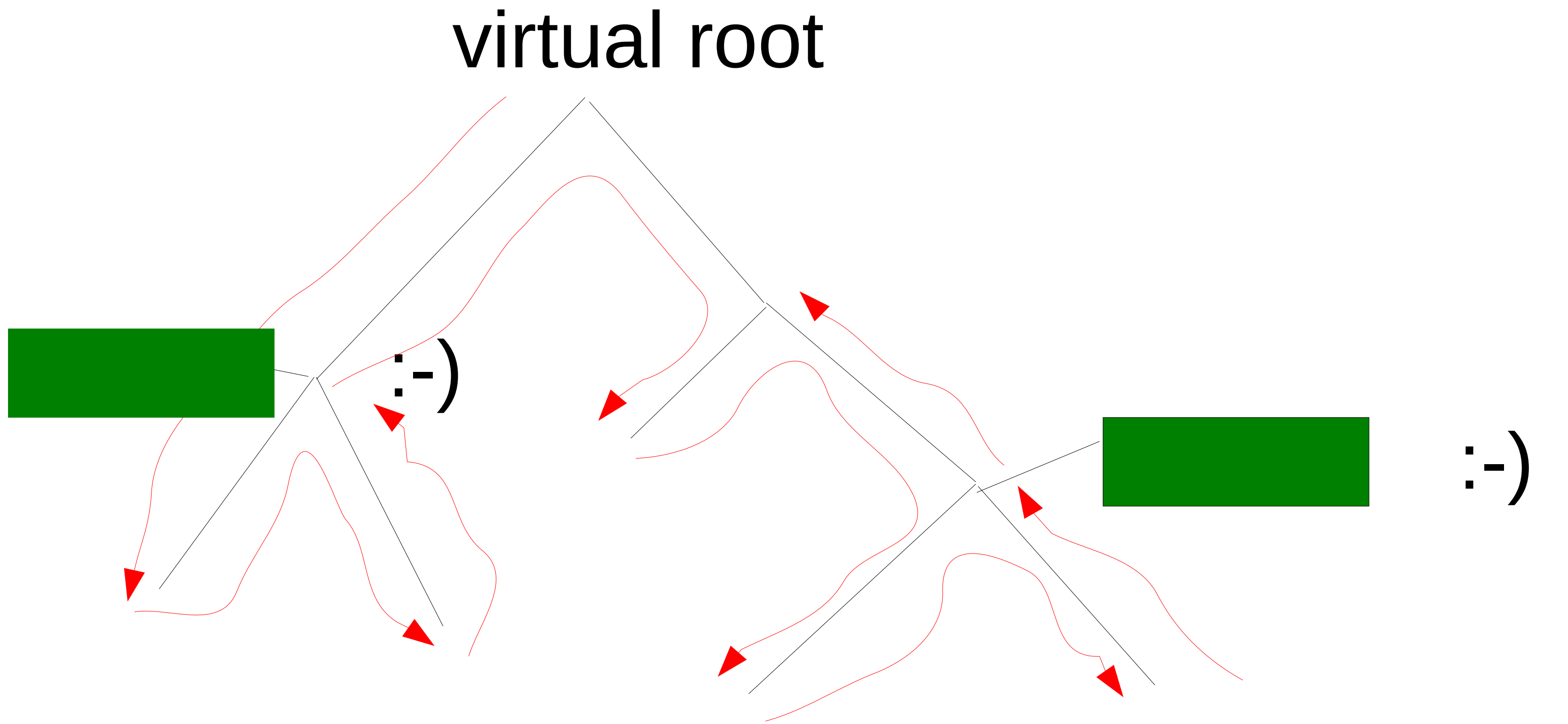
Post-order Traversal



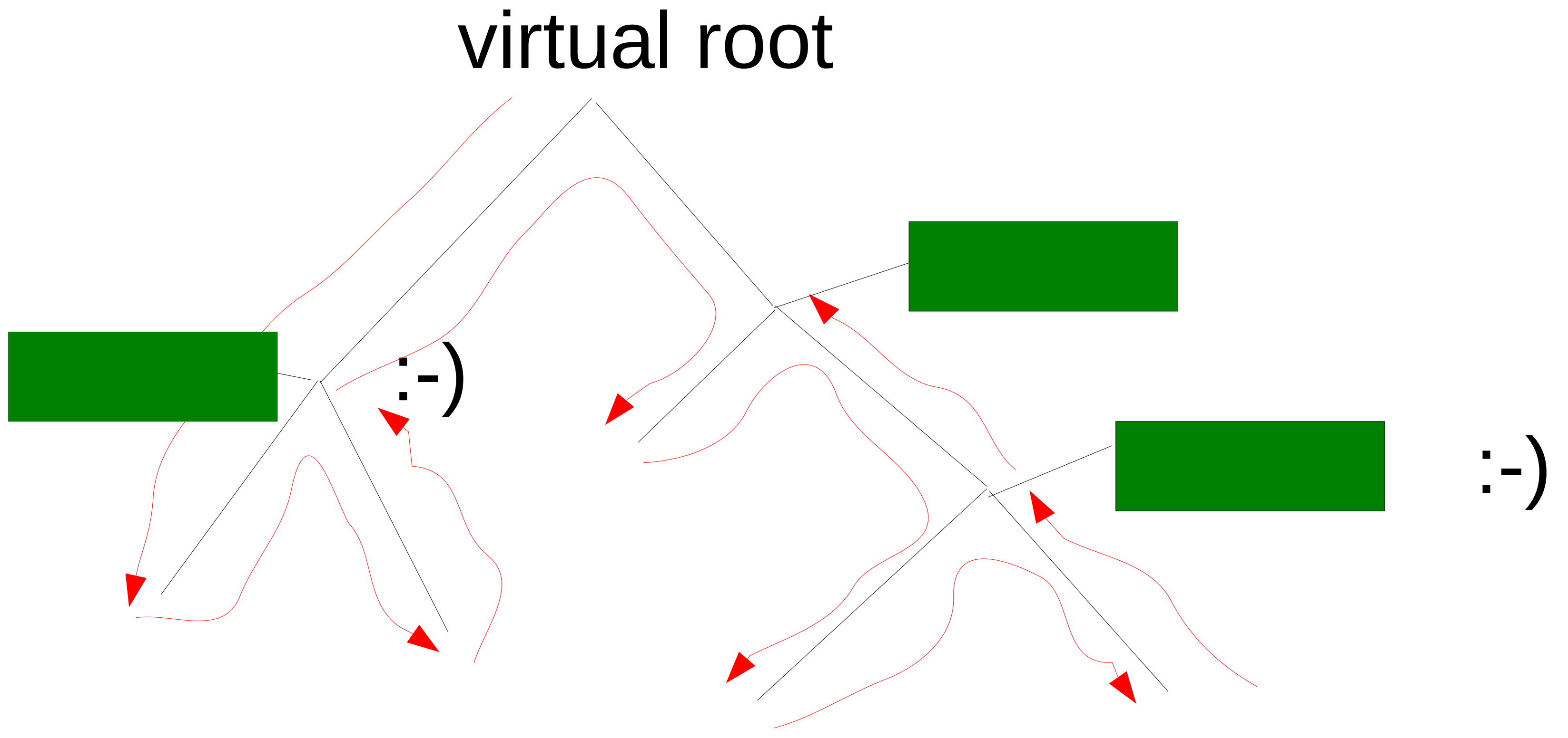
Post-order Traversal



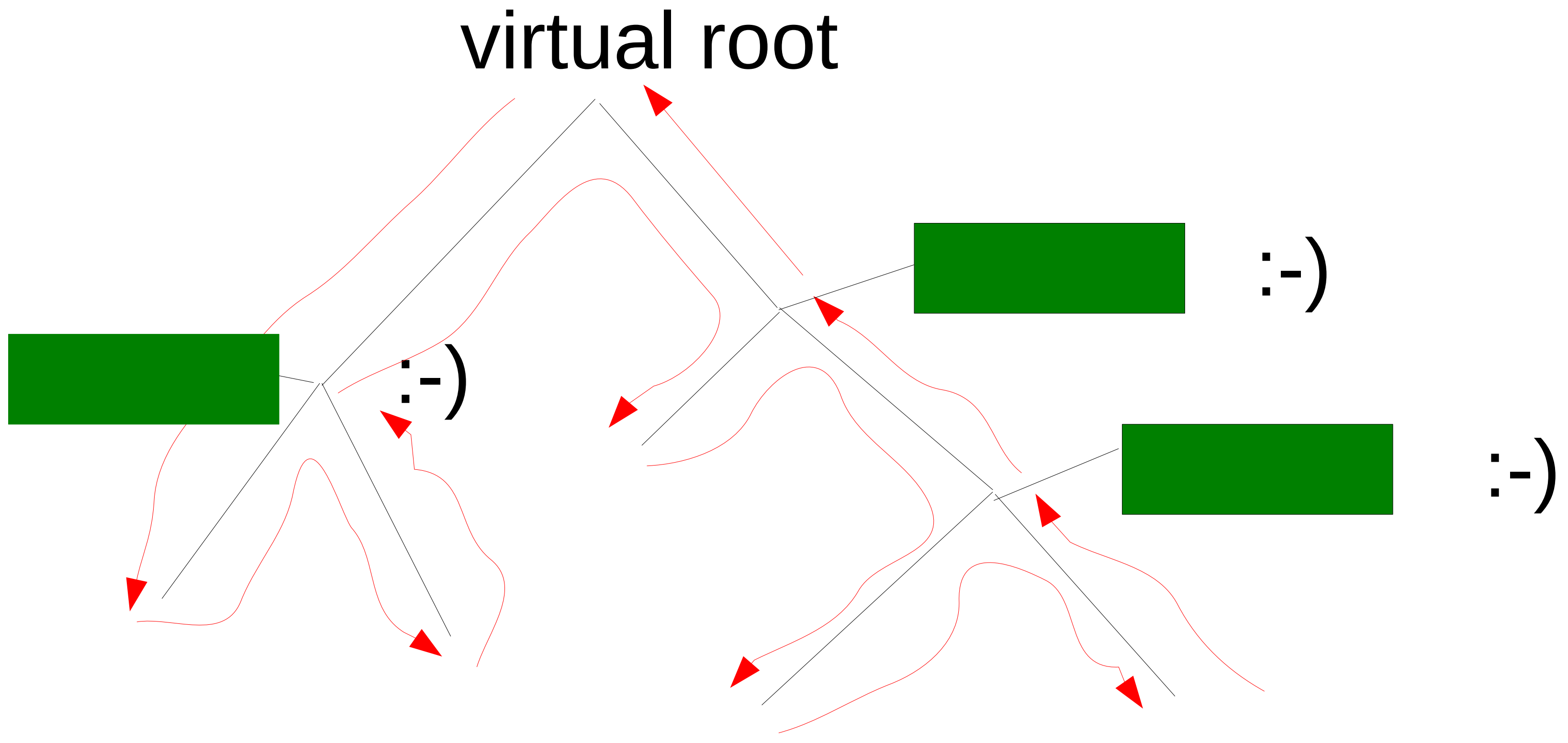
Post-order Traversal



Post-order Traversal

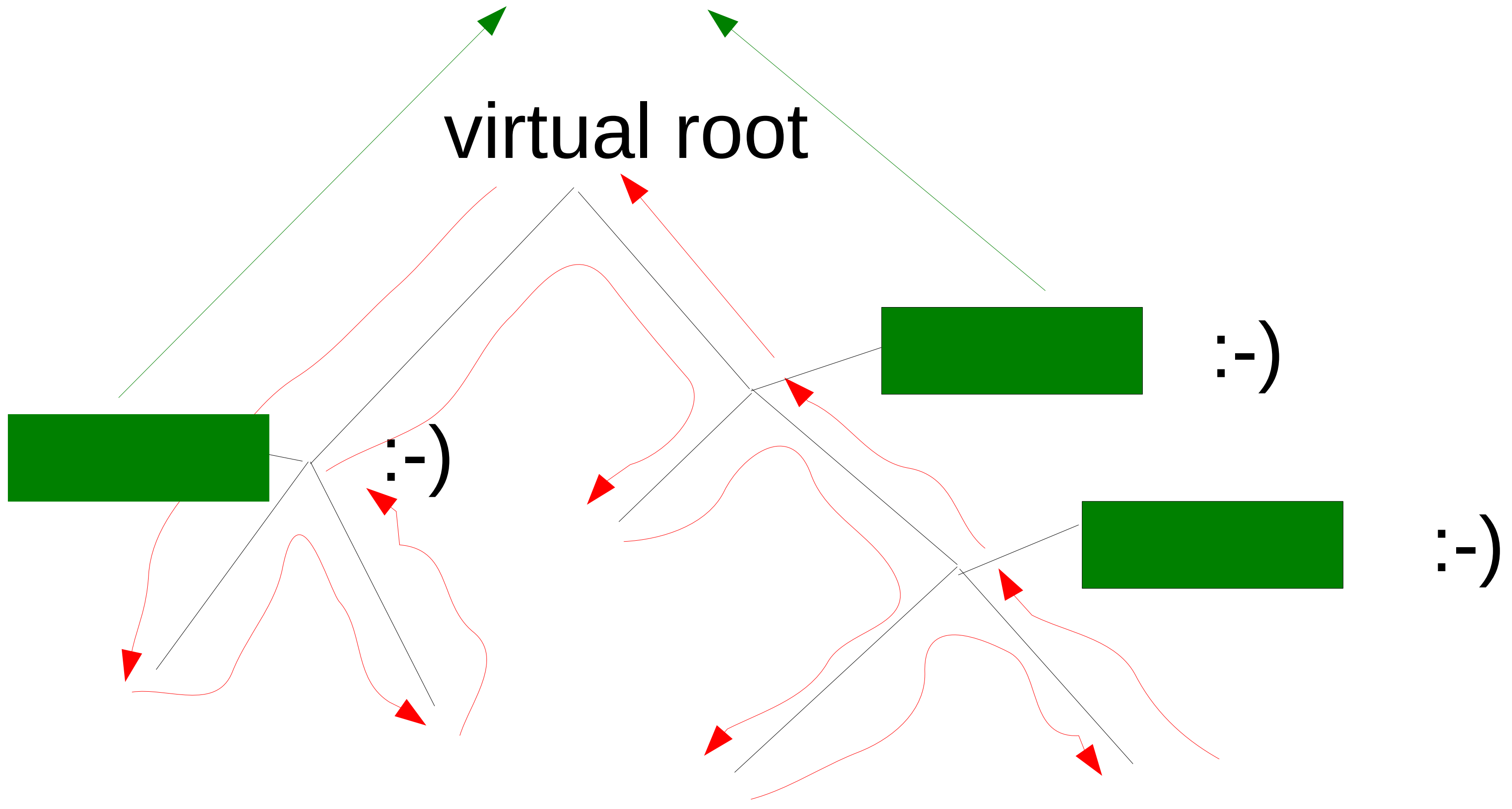


Post-order Traversal



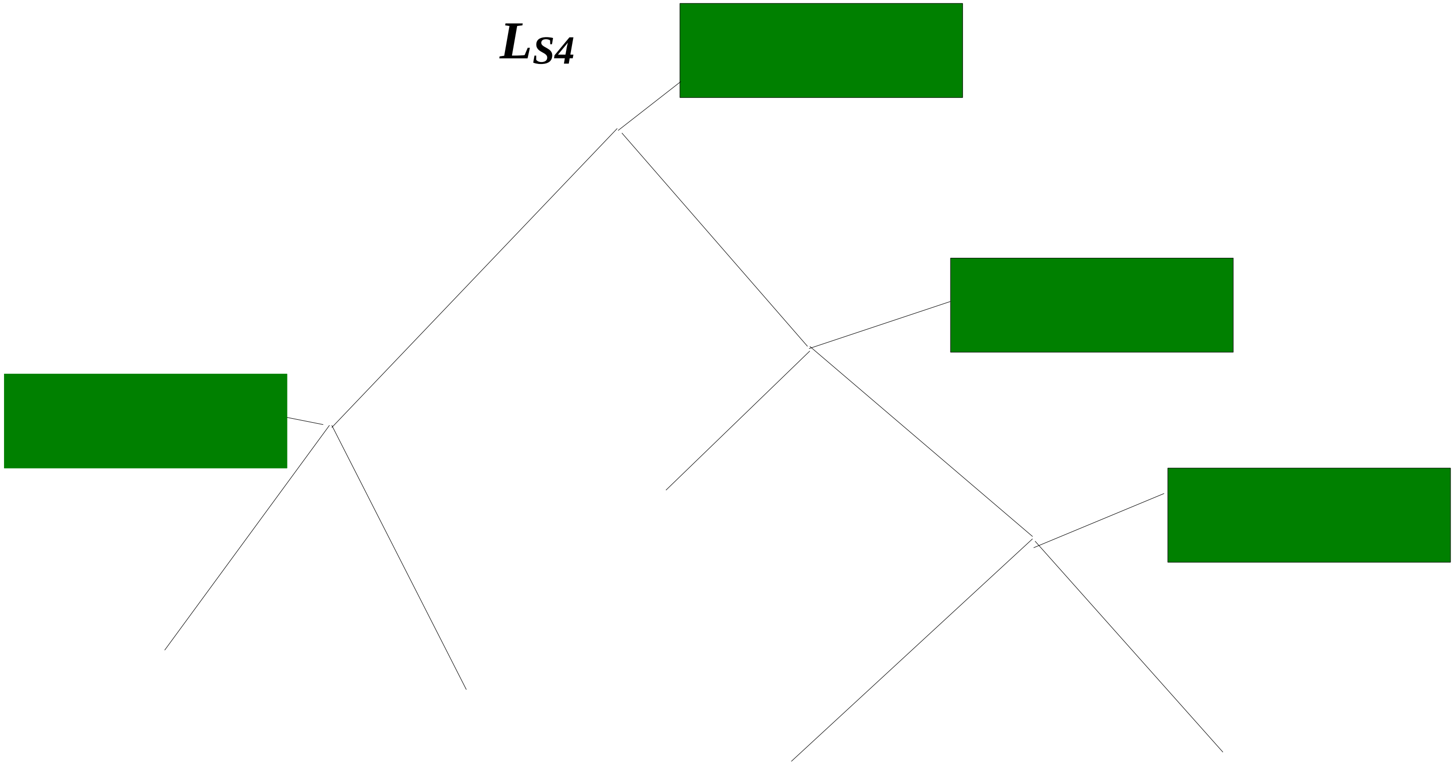
Post-order Traversal

Overall likelihood: sum over logarithms of per-site likelihoods

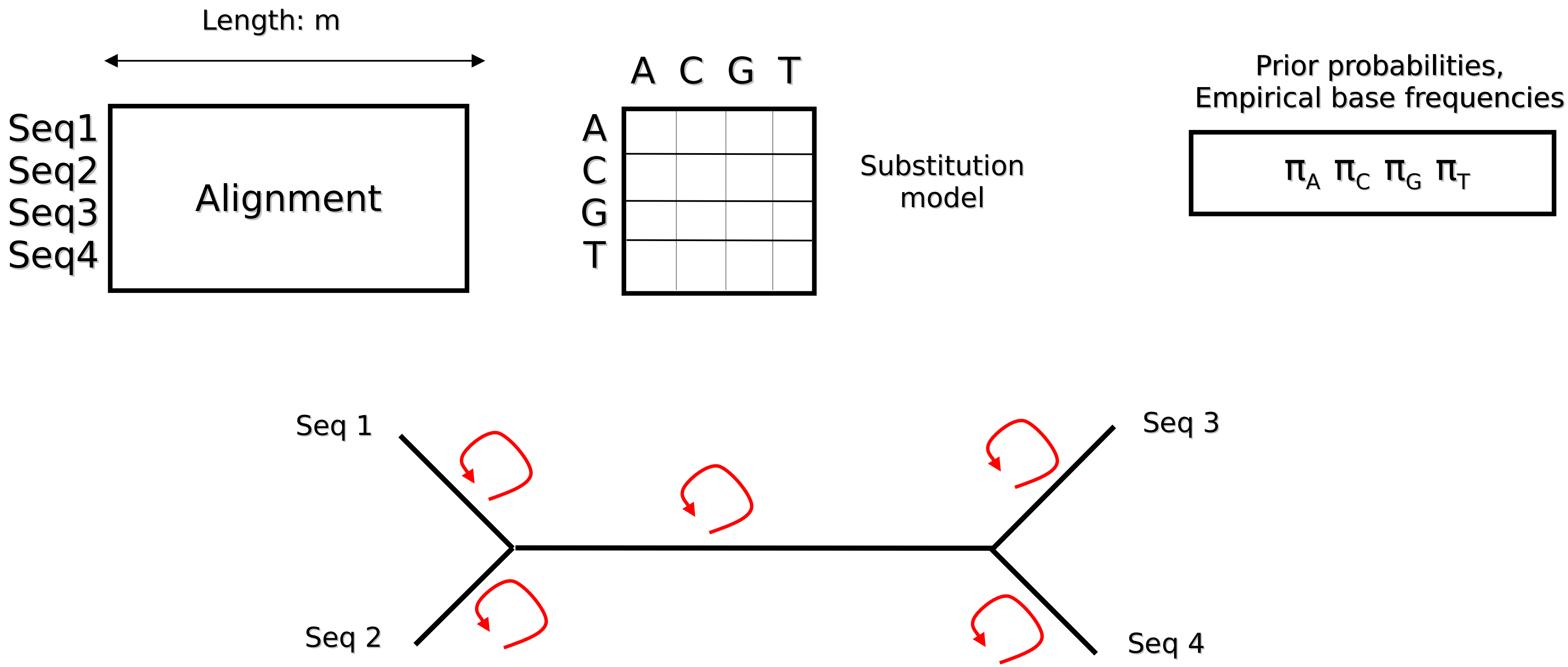


Post-order Traversal

$$L = \sum_{S_4=A}^{T} \pi_{S_4} L_{S_4}$$

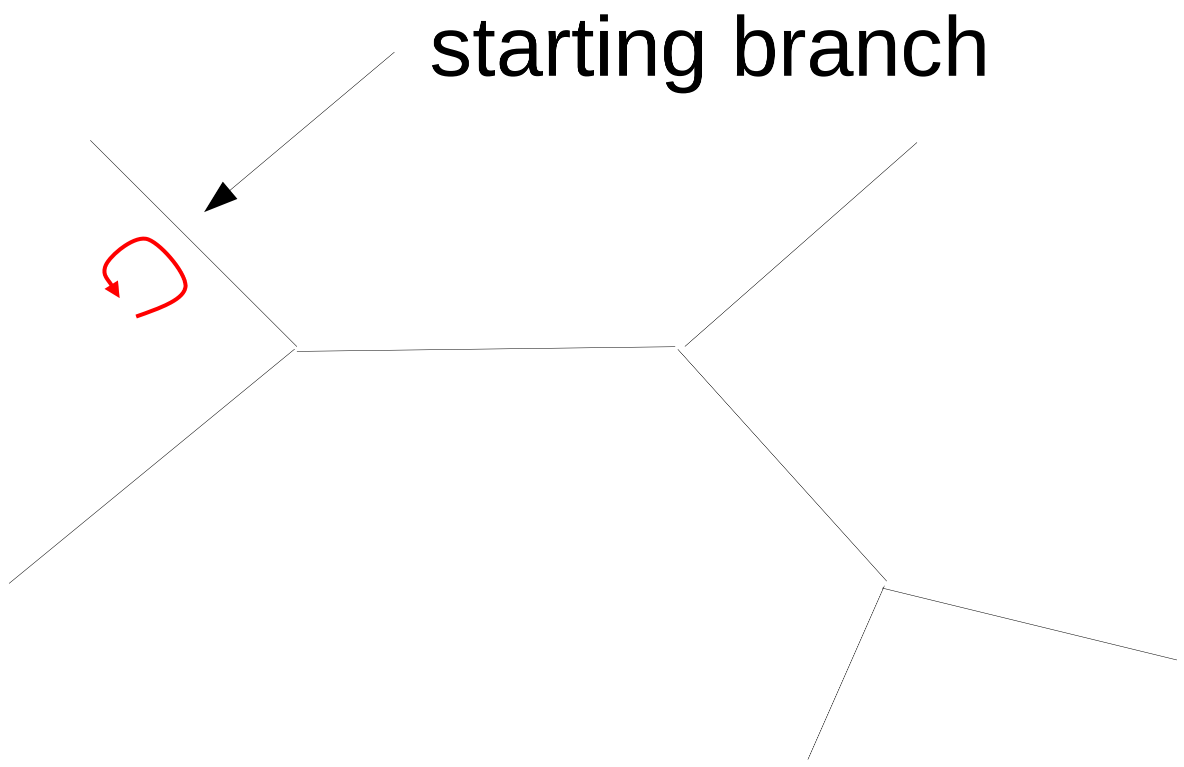


Maximum Likelihood

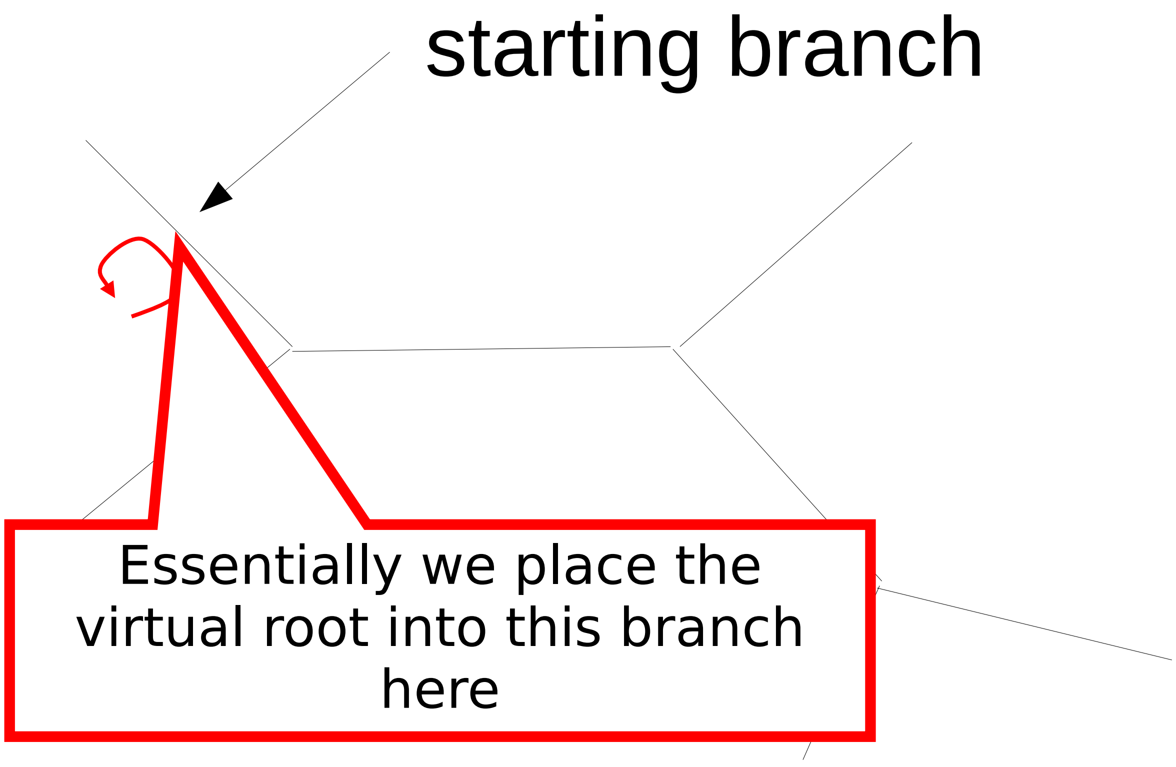


optimize branch lengths

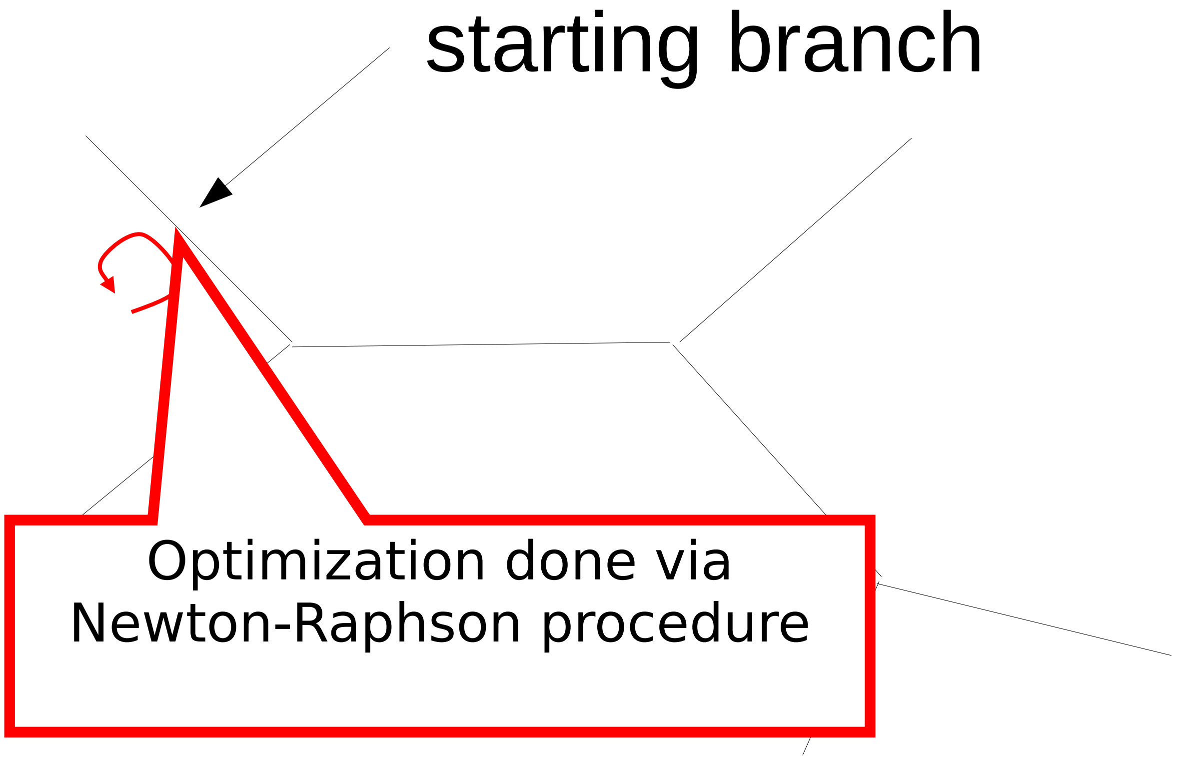
Branch Length Optimization



Branch Length Optimization



Branch Length Optimization



Newton Rapshon

- We want to find the branch length *b* that maximizes the likelihood *L(b)* of the tree
- For this, we want to know where the *first* derivative of *L(b)* is *0*
- To achieve this numerically we use the Newton-Raphson procedure for root finding deploying the first and second derivative of the likelihood *L'(b)* and *L''(b)*
- Note that, the likelihood only depends on branch *b*, all other model parameters (*Q* matrix, base frequencies, tree topology) remain fixed

Derivatives of *L(b)*

- To compute the derivatives of *L(b)*, we essentially need to be able to compute the derivatives of *P(b)* since the rest is just sums and does not depend on *b*
- Recall

  $P(b) = e^{Qb} = Ue^{\Lambda b}U^{-1}$

- thus

  $(P(b))' = U\Lambda e^{\Lambda b}U^{-1}$

- and

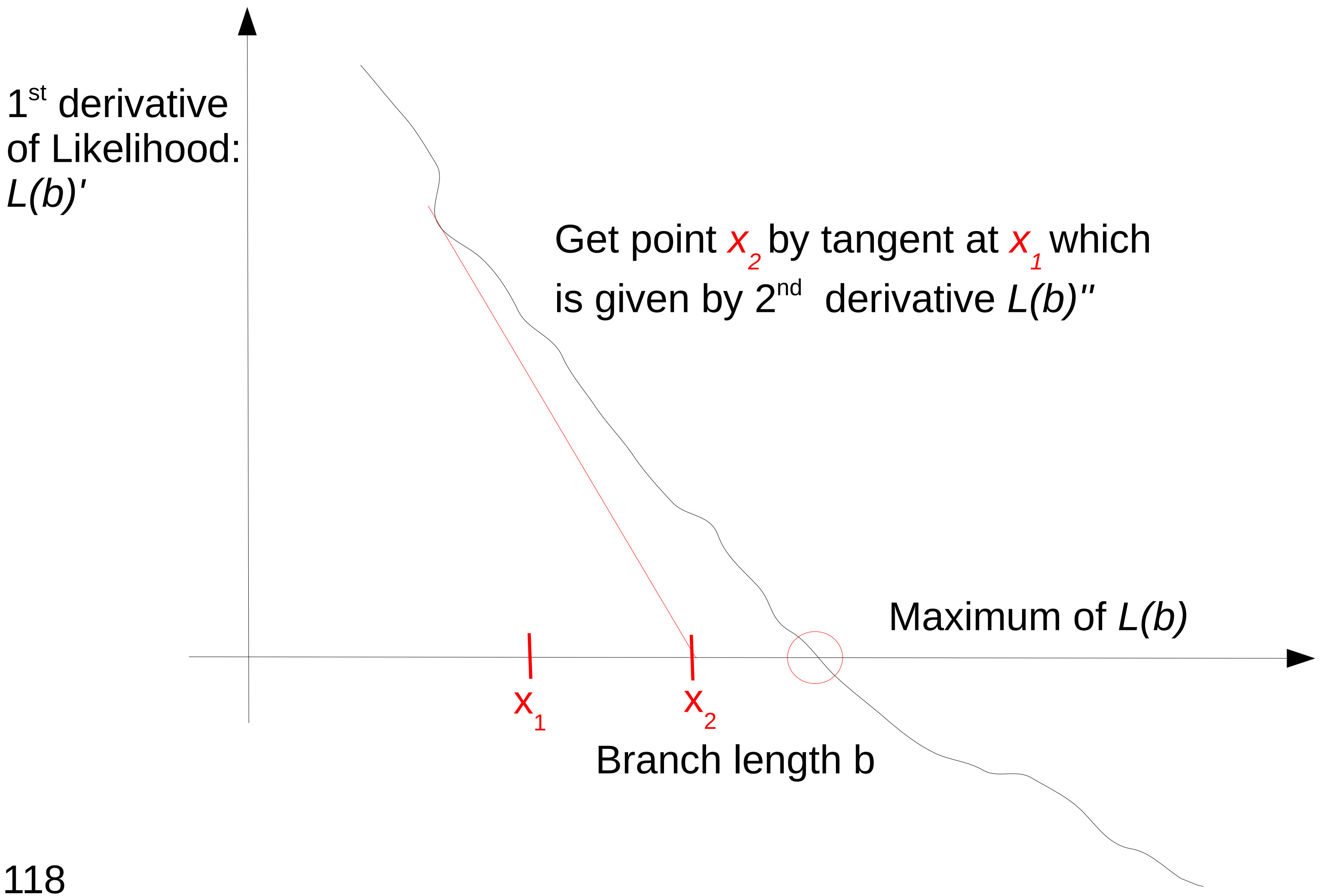
  $(P(b))'' = U\Lambda^2 e^{\Lambda b}U^{-1}$

- In practice we compute the derivatives of the log likelihood *log(L(b))*, but it is essentially the same (see next slide)

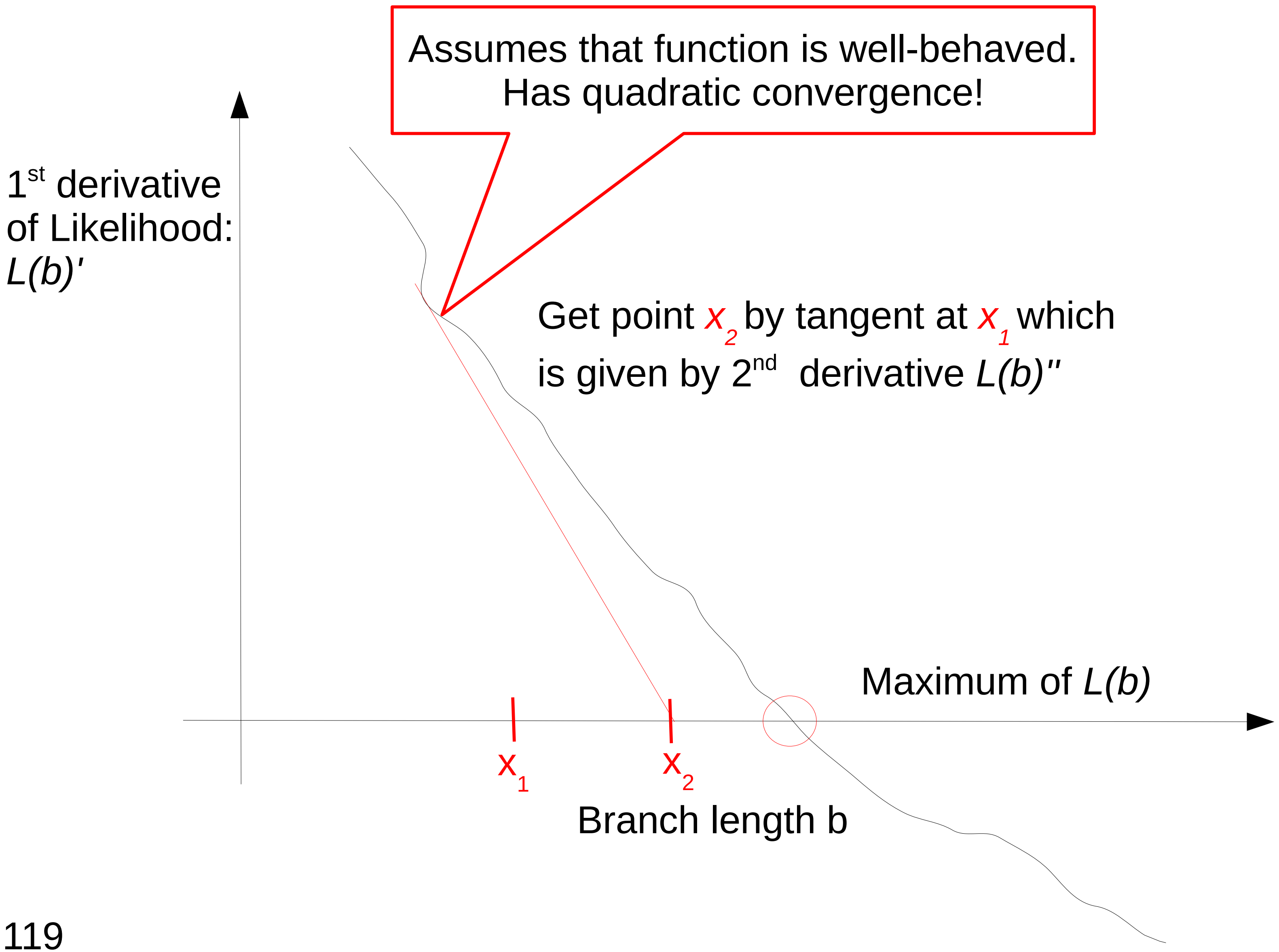
Derivatives of *log(L(b))*

- 1st derivative: $L(b)'/L(b)$
- *2nd derivative:* $(L(b)\ L(b)'' - (L(b)'')^2\ )\ /\ L(b)^2$

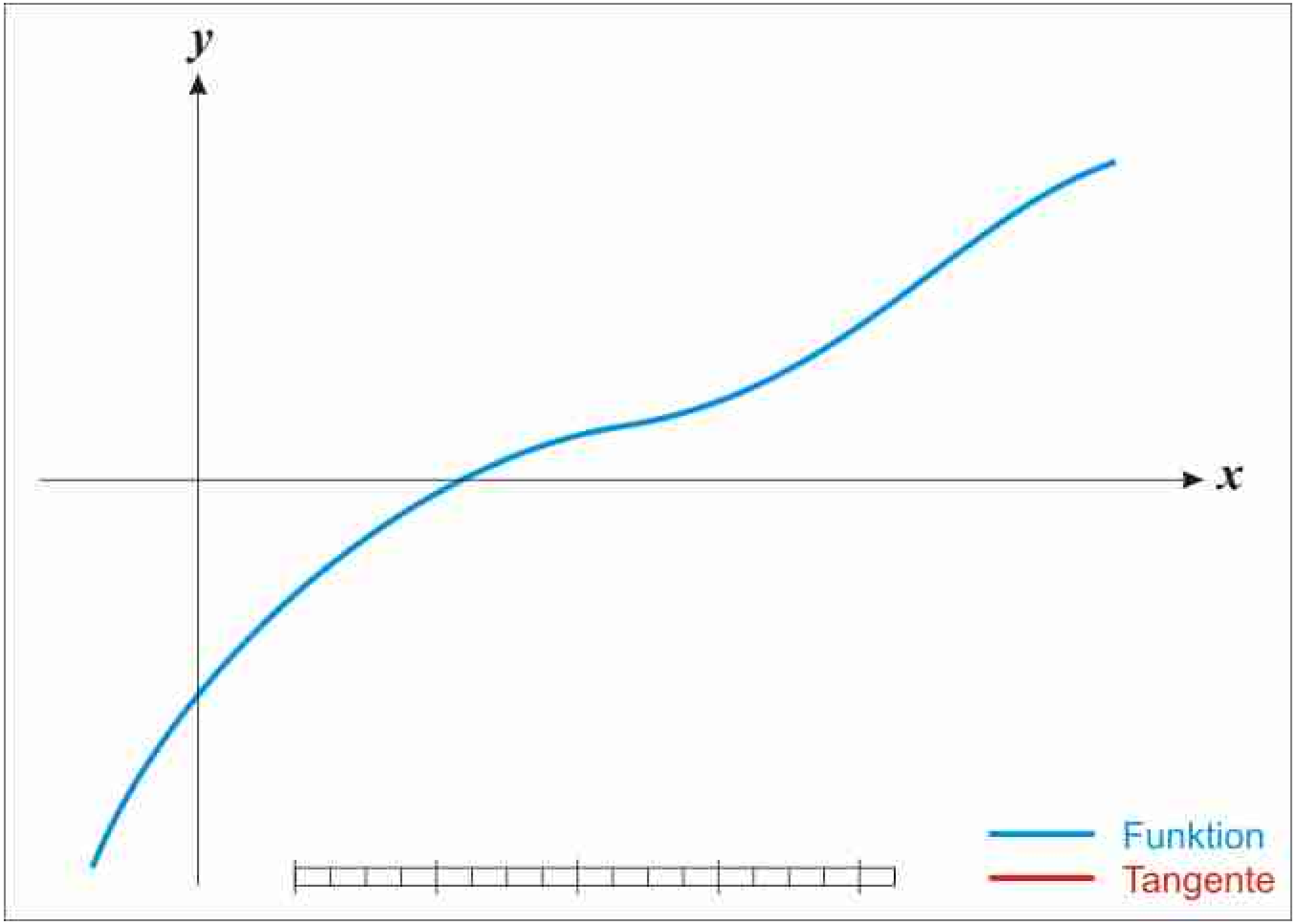
Newton Raphson



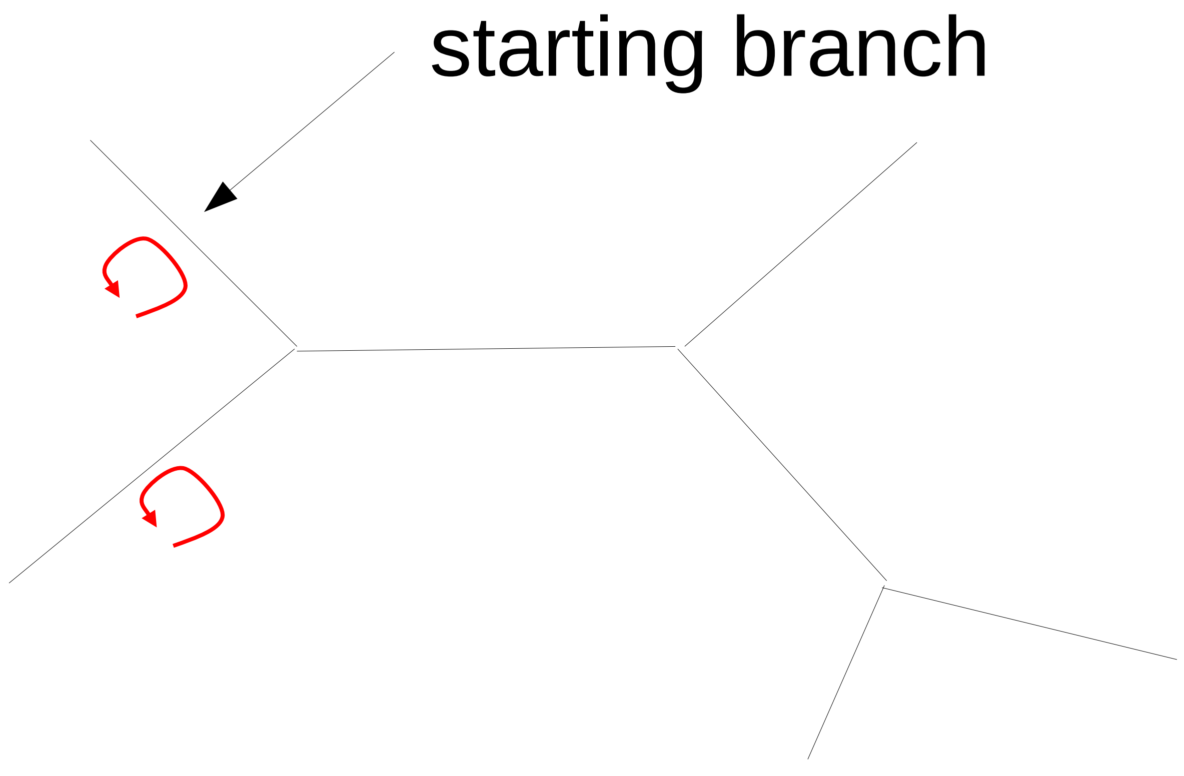
Newton Raphson



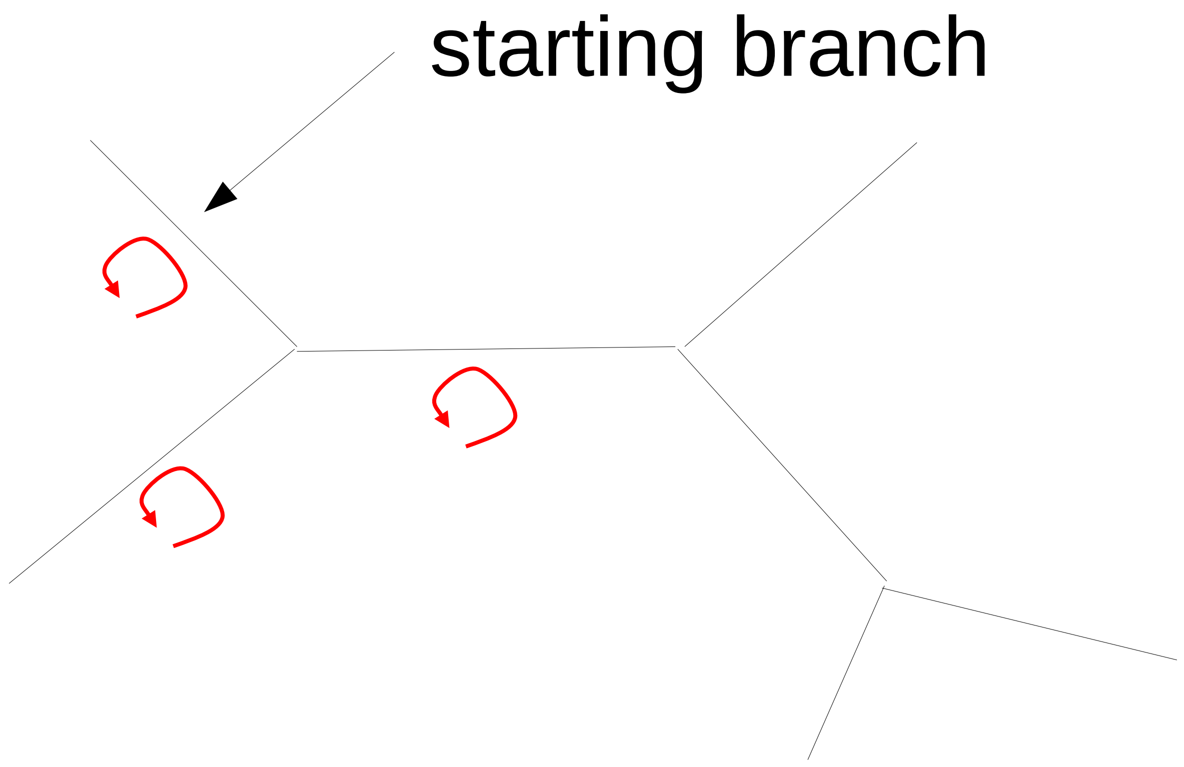
An animation



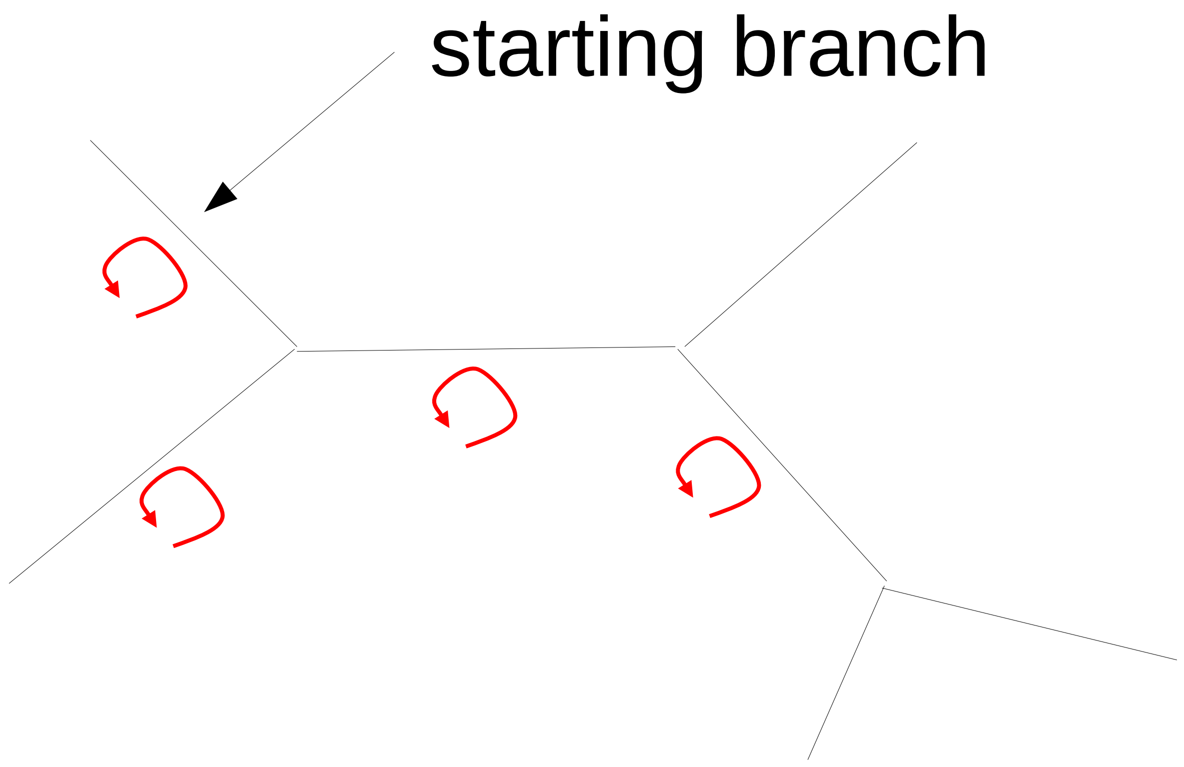
Branch Length Optimization



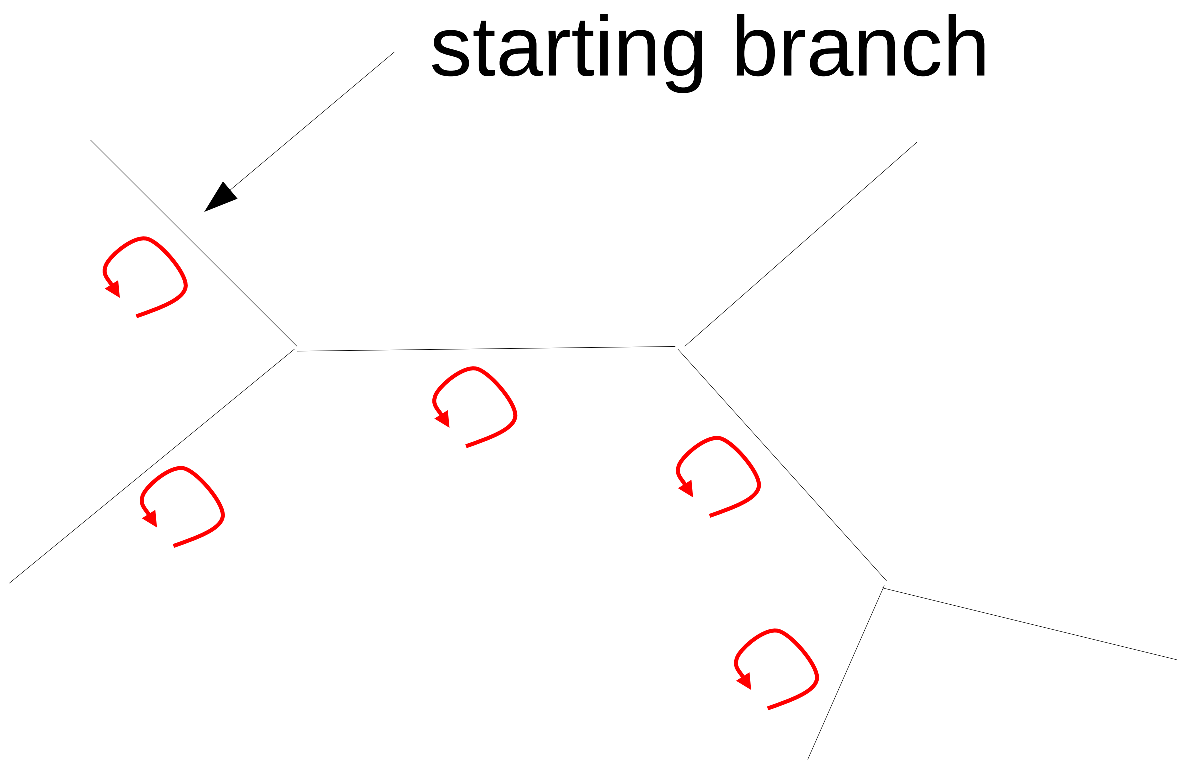
Branch Length Optimization



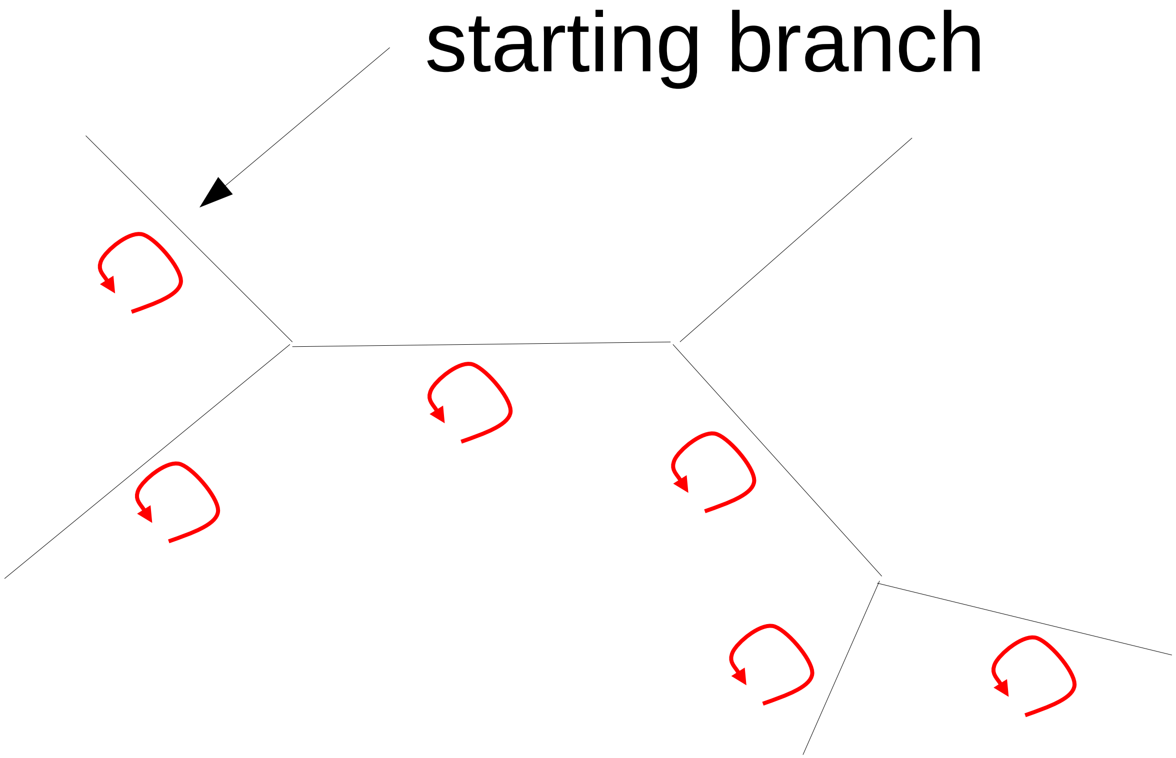
Branch Length Optimization



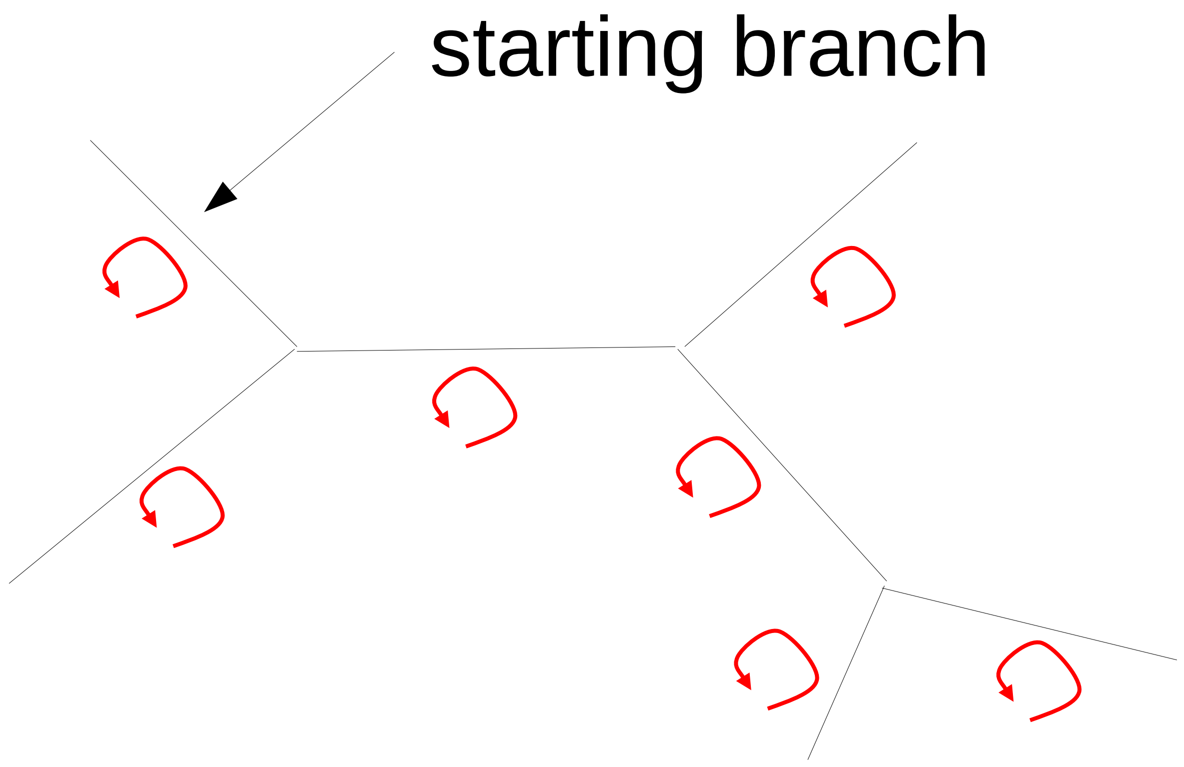
Branch Length Optimization



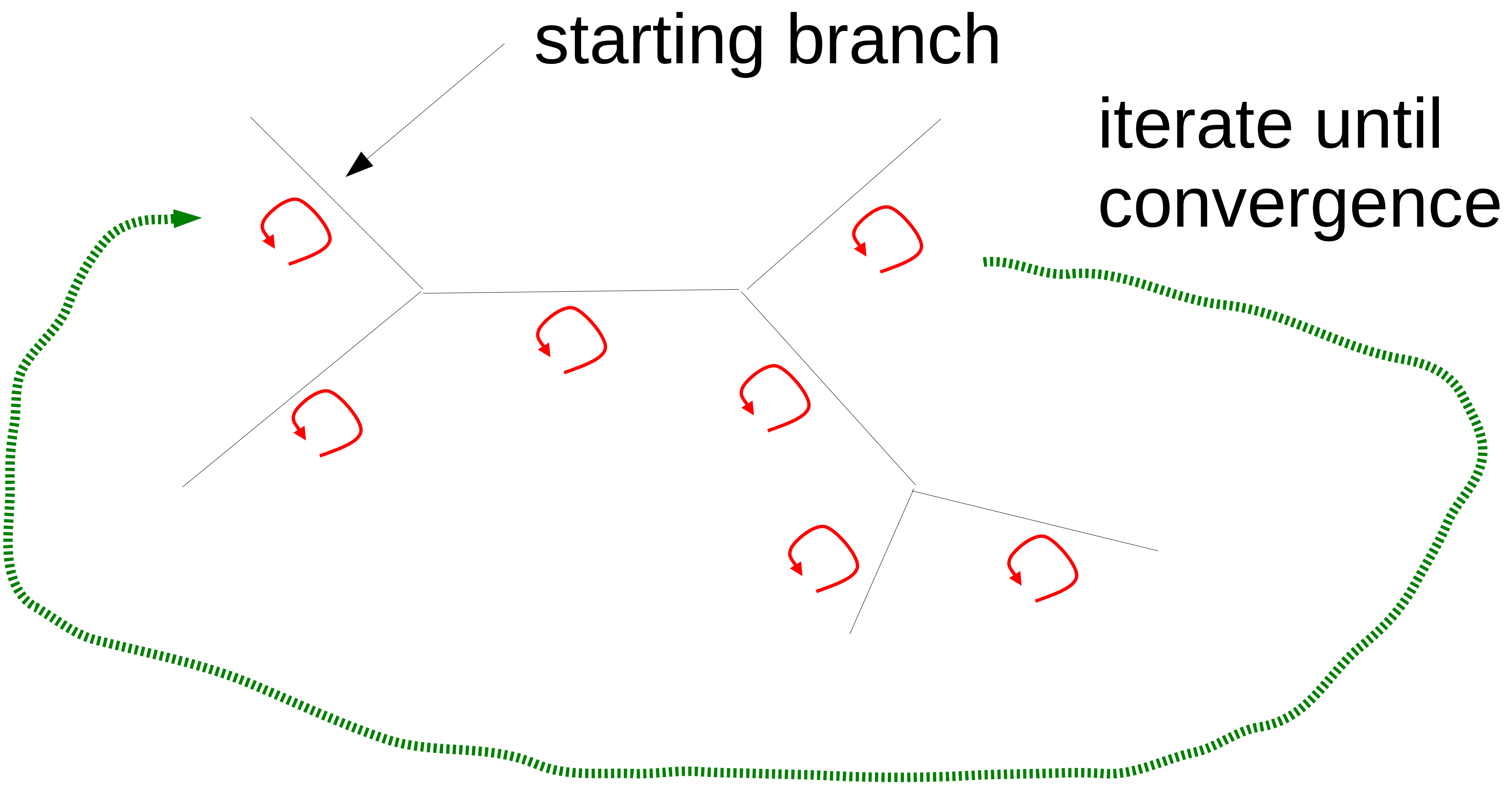
Branch Length Optimization



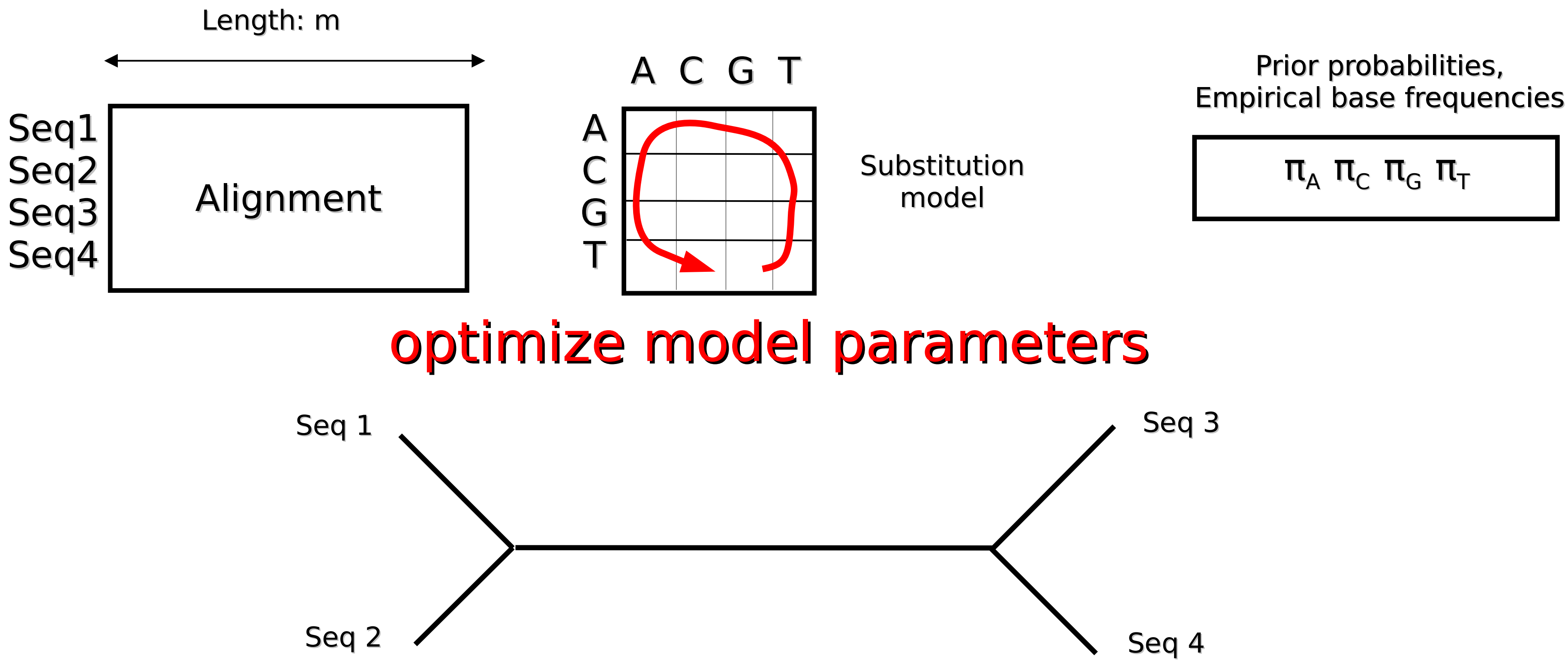
Branch Length Optimization



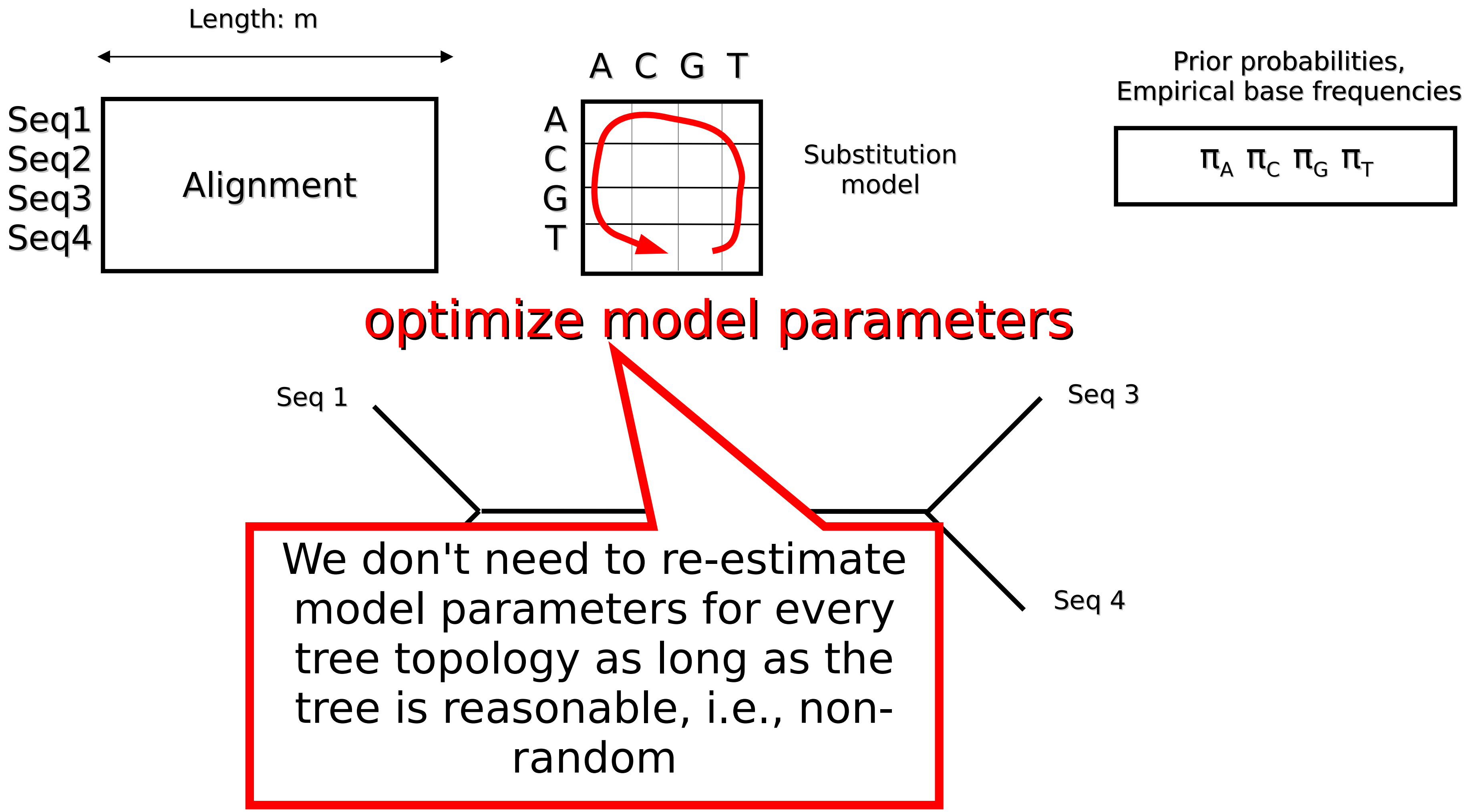
Branch Length Optimization



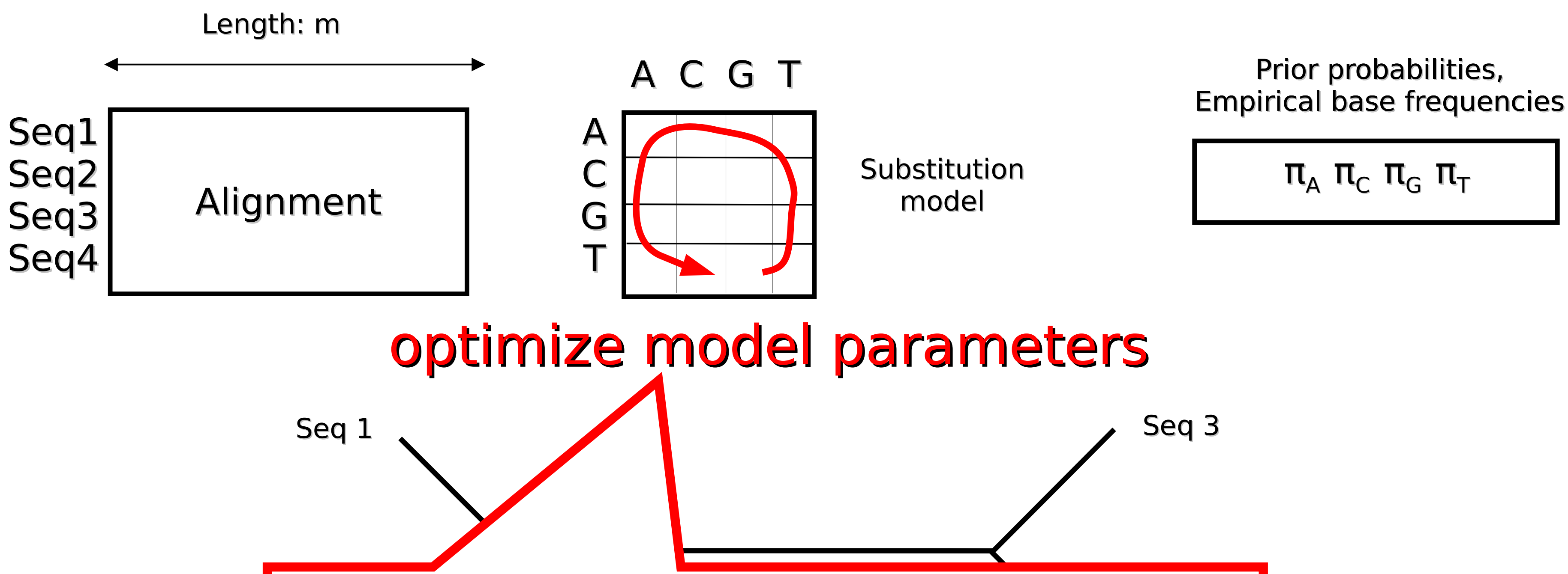
Maximum Likelihood



Maximum Likelihood



Maximum Likelihood



Methods used for model parameter optimization (other than branch lengths)

1. BFGS
2. Brent's method
3. Expectation maximization approaches

Numerical Optimization Procedures

- See chapters 9 & 10 of: *Numerical Recipes in C – The Art of Scientific Computing*

Basic Operations
Maximum Likelihood

- Compute Conditional Likelihood Vector at an inner node
- Compute Likelihood at Virtual Root
- Optimize a Branch Length for a given Branch
- Optimize all Branch Lengths
- Optimize other Model Parameters

Basic Operations
Maximum Likelihood

- Compute Conditional Likelihood Vector at an inner node
- Compute Likelihood at Virtual Root
- Optimize a Branch Length for a given Branch
- Optimize all Branch Lengths
- Optimize other Model Parameters

The optimizers are the tricky routines!