CS 364 COMPUTATIONAL BIOLOGY

Sara Mathieson
Haverford College

Outline

- Limitations of parsimony
- Likelihood framework for tree inference
- Bayesian phylogenetics

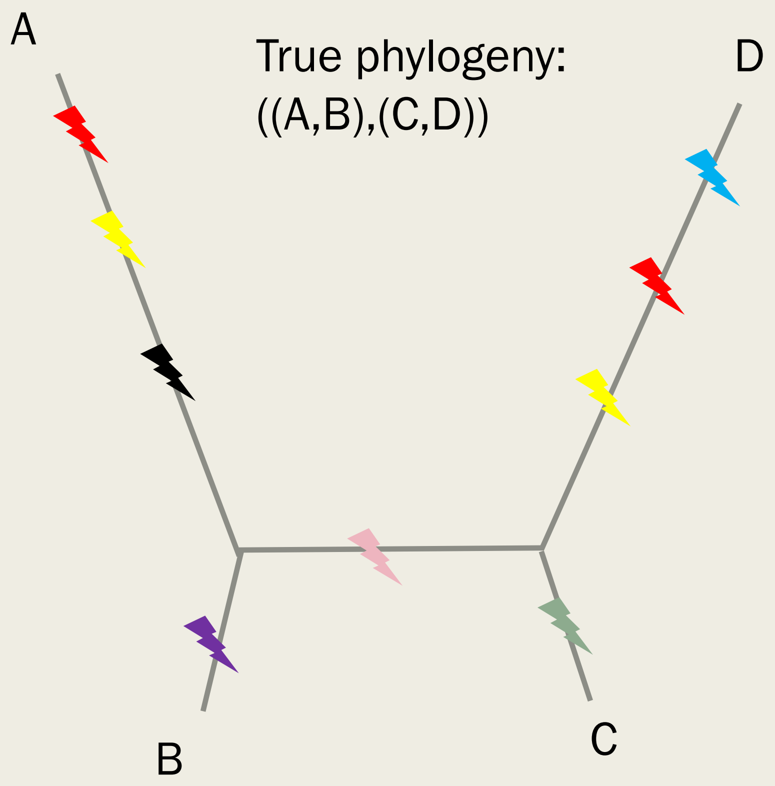
Limitations of Parsimony

Problems with parsimony

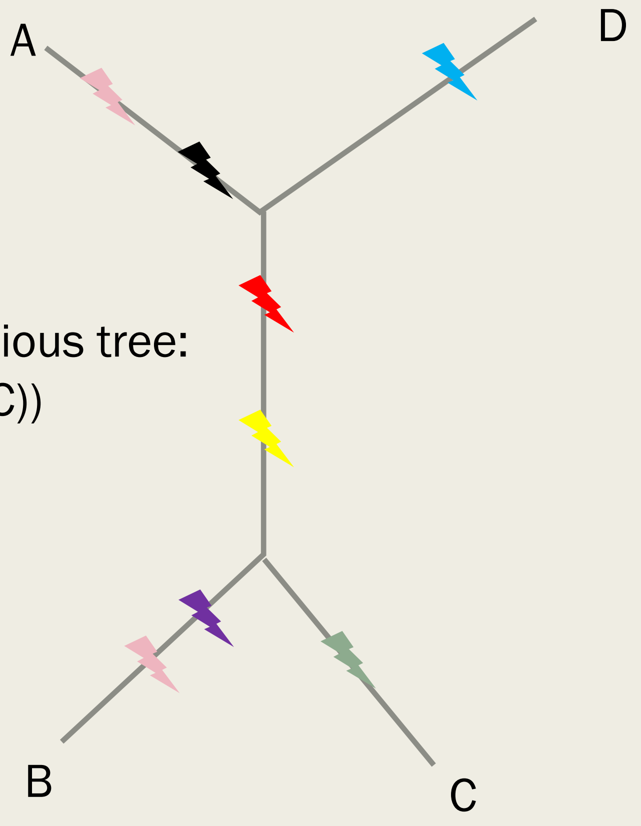
- Impractical (except for special cases – exact solution is NP-complete).
- Scales linearly with number of characters – going to be a problem for genomic data!
- Treats all characters the same – but some characters are more important than others
- Assumes convergent evolution is rare and that all mutations are equally likely
- Can be inconsistent – converges to the wrong answer when you have lots of data (long branch attraction)

Long branch attraction

If mutations happen at random, then long branches in the tree will tend to have more mutations -> they will look more similar -> they will be "attracted" to each other.



True phylogeny:
((A,B),(C,D))



Parsimonious tree:
((A,D),(B,C))

More problems with parsimony:

- Mutational “costs” are not represented in terms of measurable quantities.

- Does not use all the information in the data (e.g. does not use information at non-variable characters).

- No statistical guarantees. No estimate of uncertainty.

A possible solution – Maximum likelihood methods:

- Cast problem in terms of probabilities (e.g. 1% chance that a base mutates in one generation).

- Uses all information in the data.

- Efficient, [more] consistent, accounts properly for repeat and convergent evolution.

- Can measure uncertainty

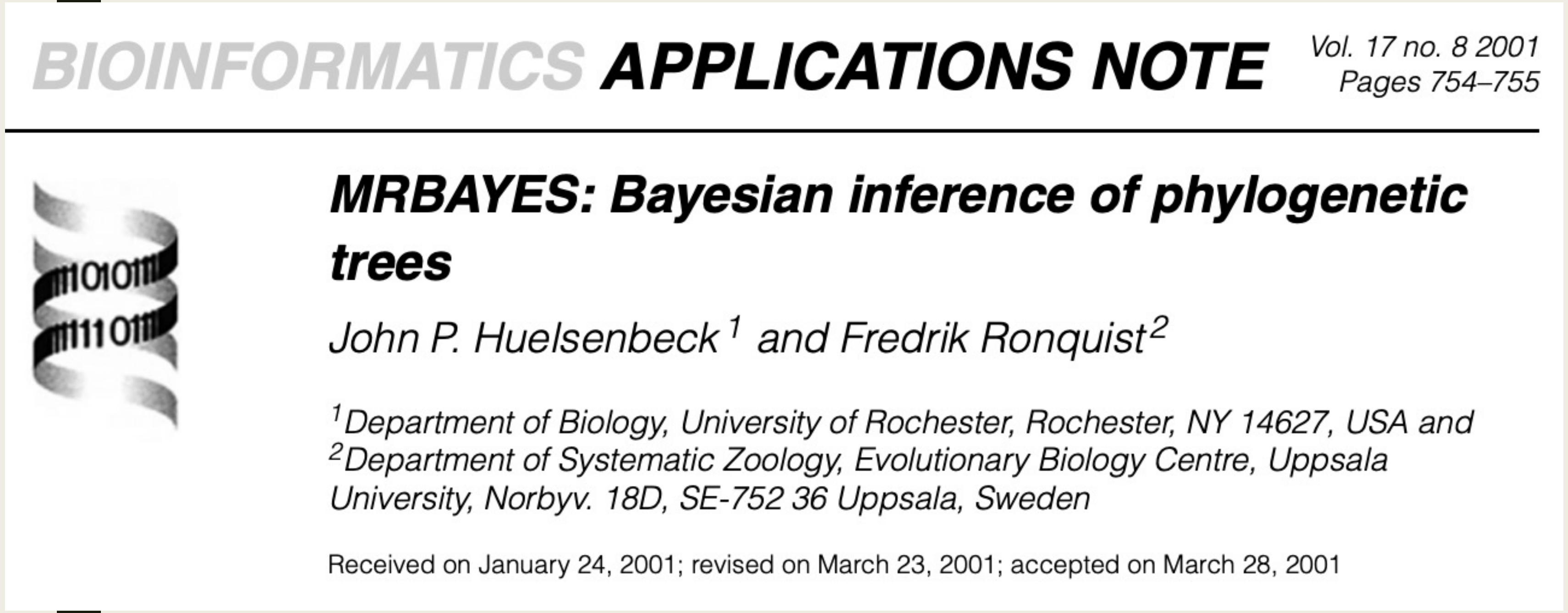
Likelihood framework for tree inference

Likelihood tells you **how surprised you should be** at the observed data

High likelihood ⇨ **less** surprised

Low likelihood ⇨ **more** surprised

MrBayes (26,000 citations and counting)



BIOINFORMATICS APPLICATIONS NOTE Vol. 17 no. 8 2001 Pages 754–755

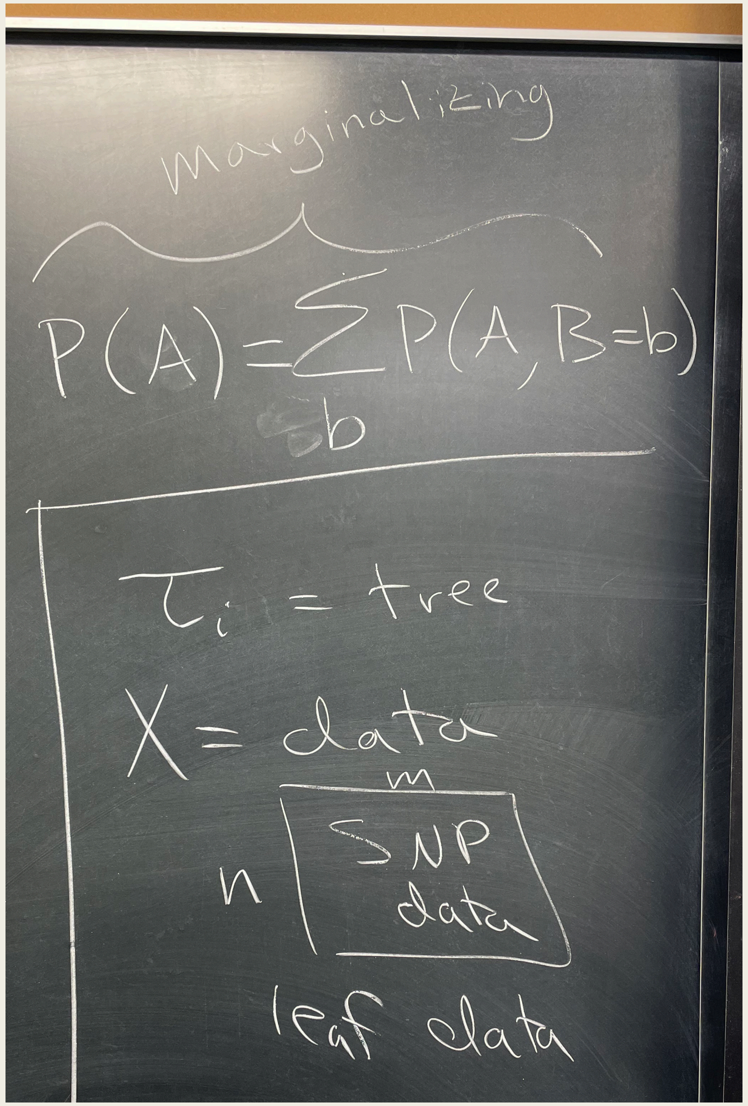
MRBAYES: Bayesian inference of phylogenetic trees

John P. Huelsenbeck[1] and Fredrik Ronquist[2]

[1]Department of Biology, University of Rochester, Rochester, NY 14627, USA and [2]Department of Systematic Zoology, Evolutionary Biology Centre, Uppsala University, Norbyv. 18D, SE-752 36 Uppsala, Sweden

Received on January 24, 2001; revised on March 23, 2001; accepted on March 28, 2001

$$f(\tau_i|\boldsymbol{X}) = \frac{f(\boldsymbol{X}|\tau_i)f(\tau_i)}{\sum_{j=1}^{B(s)} f(\boldsymbol{X}|\tau_j)f(\tau_j)}$$



$$\overbrace{P(A)=\sum_b P(A,B=b)}^{\text{marginalizing}}$$

τ_i = tree

X = data

$$n\ \overset{m}{\boxed{\text{SNP data}}}$$

leaf data

Recall 260!

$$\underbrace{P(\text{tree} \mid \text{data})}_{\text{posterior}} = \frac{P(\text{tree}, \text{data})}{P(\text{data})}$$

$$= \frac{\overbrace{P(\text{tree})}^{\text{prior}}\,\overbrace{P(\text{data} \mid \text{tree})}^{\text{likelihood}}}{\underbrace{P(\text{data})}_{\text{evidence}}}$$

$$\boxed{P(A,B) = P(A)P(B \mid A)}$$

Bayes Rule

$$= \frac{P(\text{tree})\,P(\text{data} \mid \text{tree})}{\sum_{\text{tree}'} P(\text{tree}')\,P(\text{data} \mid \text{tree}')}$$

$\uparrow$ all possible trees.

Likelihood of a single vertex

First 32 nucleotides of the ψη-globin gene of gorilla:

● **GAAGTCCTTGAGAAATAAACTGCACACACTGG**

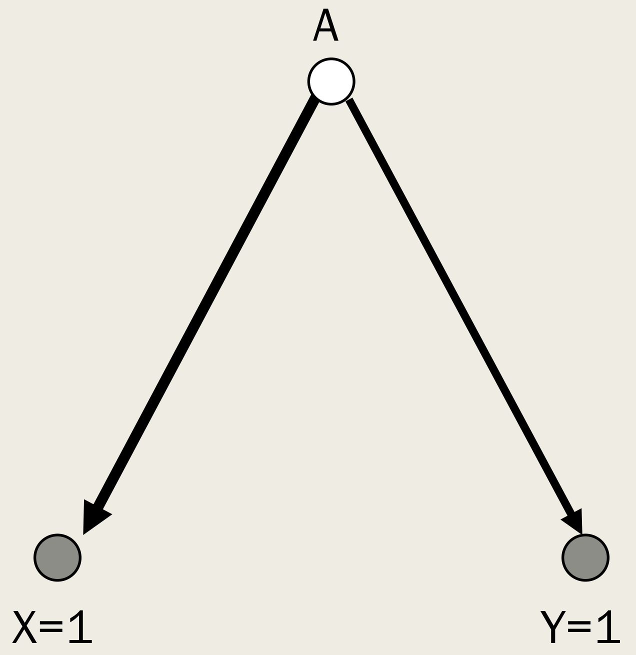
$$L = \Pr(G)\ \Pr(A)\ \Pr(A)\ \Pr(G)\ \Pr(T) \qquad \Pr(G)$$

$$L = \pi_G\ \pi_A\ \pi_A\ \pi_G\ \pi_T \qquad \pi_G$$

$$L = \pi_A^{12}\ \pi_C^{7}\ \pi_G^{7}\ \pi_T^{6}$$

$$\log L = 12\log(\pi_A) + 7\log(\pi_C) + 7\log(\pi_G) + 6\log(\pi_T)$$

Computing likelihoods of tree states



Simple model: We see two sequences today

What is A?

Mutation rate matrix:

	0	1
0	0.8	0.2
1	0.1	0.9

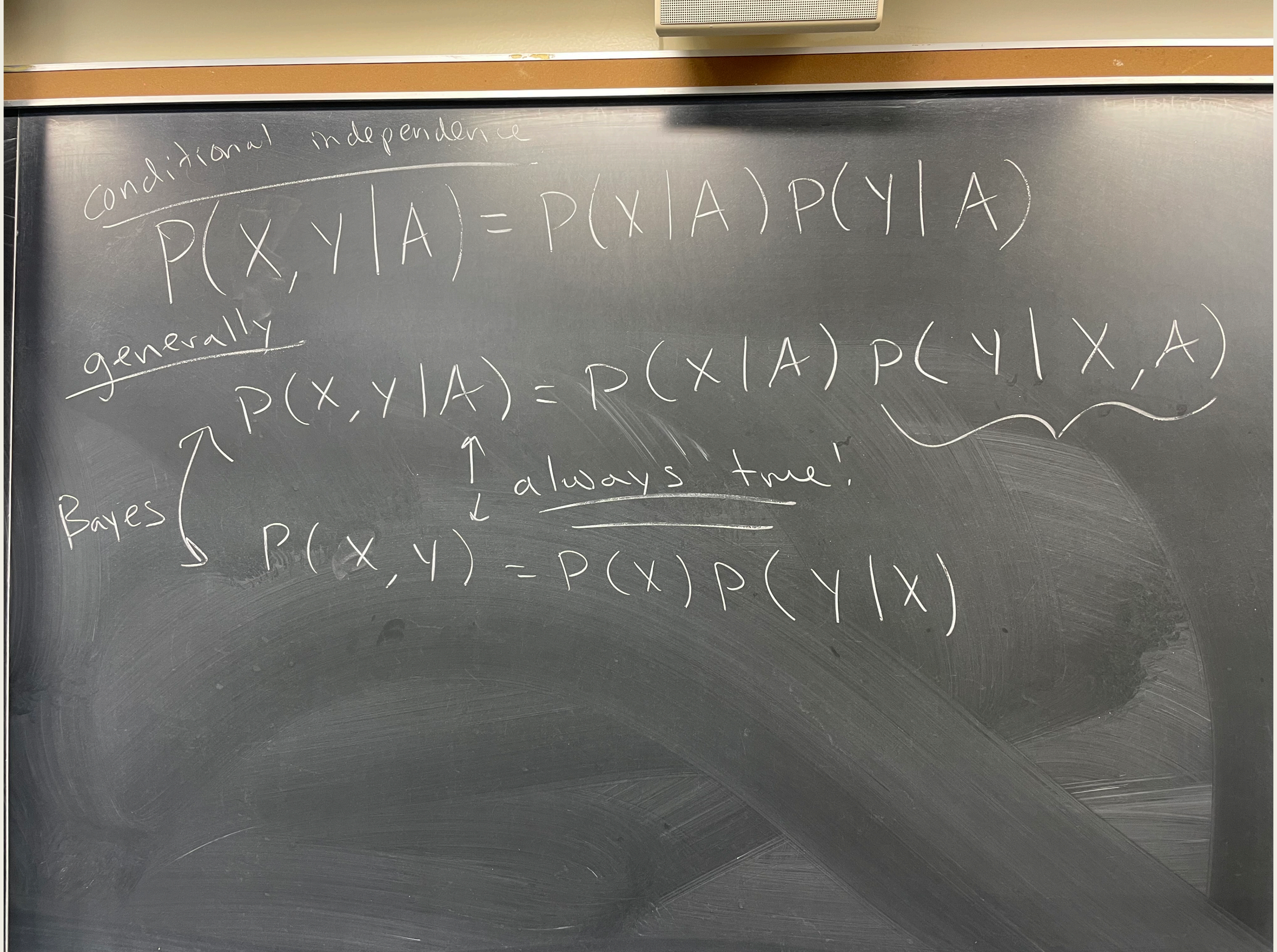
Either A=0 or A=1

Now we are treating the state at A as the parameter, so look at $\ell(\mathrm{A}) = \mathrm{P}(\mathrm{data}|\mathrm{A})$

$$\ell(0) = \mathrm{P}(\mathrm{X} = 1, \mathrm{Y} = 1|\mathrm{A} = 0) = P(X = 1|A = 0)P(Y = 1|A = 0) = 0.2 * 0.2 = 0.04$$

$$\ell(1) = \mathrm{P}(\mathrm{X} = 1, \mathrm{Y} = 1|\mathrm{A} = 1) = P(X = 1|A = 1)P(Y = 1|A = 1) = 0.9 * 0.9 = 0.81$$

Evolution on each branch is independent!

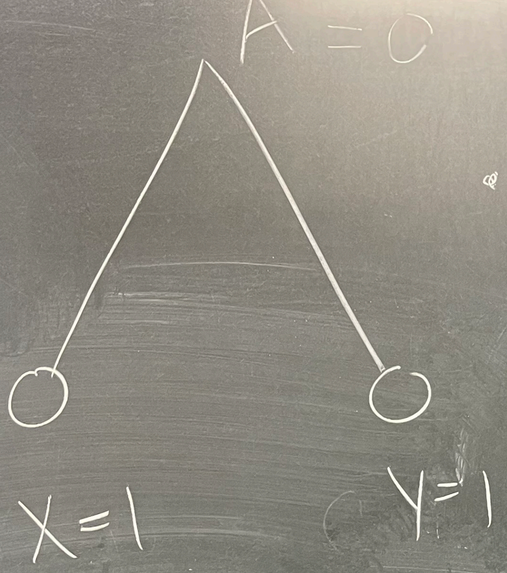


conditional independence

$$P(X,Y|A) = P(X|A)\,P(Y|A)$$

generally

$$\begin{tikzcd}
P(X,Y|A) \arrow[d, leftrightarrow, bend right, "\text{Bayes}"'] & = \arrow[d, leftrightarrow, "\text{always true!}"] & P(X|A)\,\underbrace{P(Y|X,A)} \\
P(X,Y) & = & P(X)\,P(Y|X)
\end{tikzcd}$$

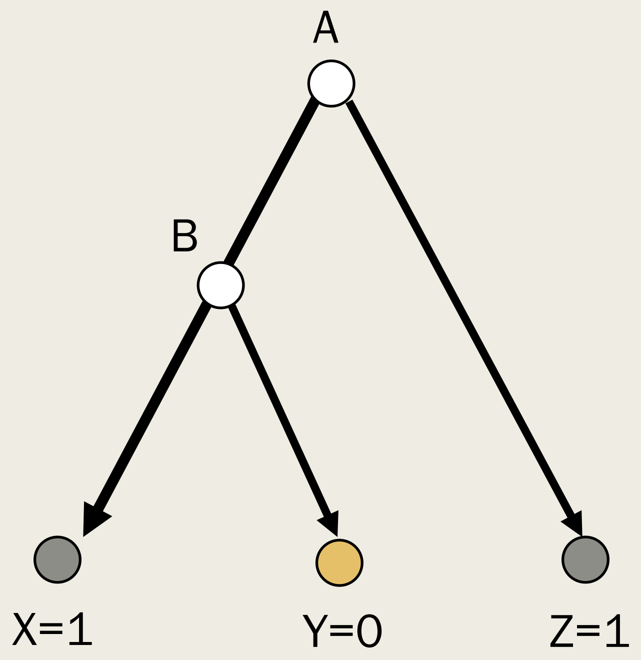


X & Y are conditionally independent given A

X & Y are not independent

$$\underbrace{\cancel{P(X,Y) = P(X)P(Y)}}_{\text{independence}}$$

Computing likelihoods of tree states



Simple model: We see three sequences today

What are A and B?

Mutation rate matrix:

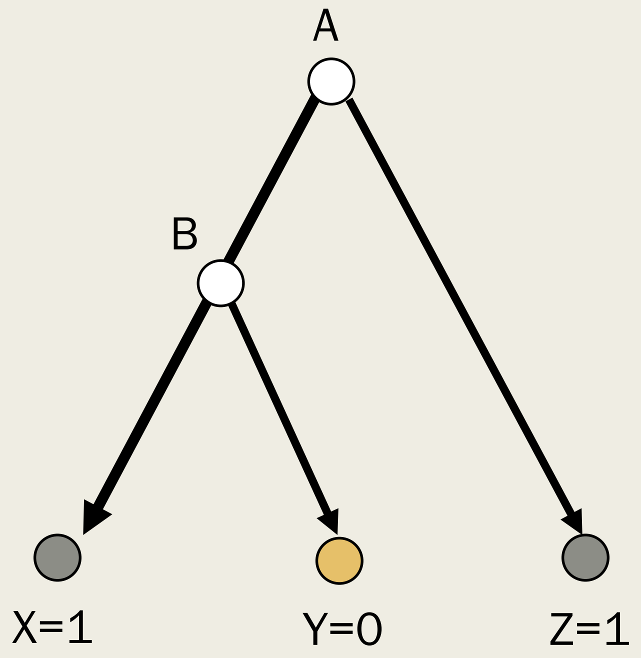
	0	1
0	0.8	0.2
1	0.1	0.9

A can be 0,1 and B can be 0,1

Now A and B parameters, so look at $\ell(\mathrm{A,B}) = \mathrm{P(X,Y,Z|A,B)} = \mathrm{P(X|B)P(Y|B)P(Z|A)}$

$\ell(0,0) = \mathrm{P}(1,0,1|0,0) =$
$\ell(0,1) = \mathrm{P}(1,0,1|0,1) =$
$\ell(1,0) = \mathrm{P}(1,0,1|1,0) =$
$\ell(1,1) = \mathrm{P}(1,0,1|1,1) =$

Computing likelihoods of tree states



Simple model: We see three sequences today

What are A and B?

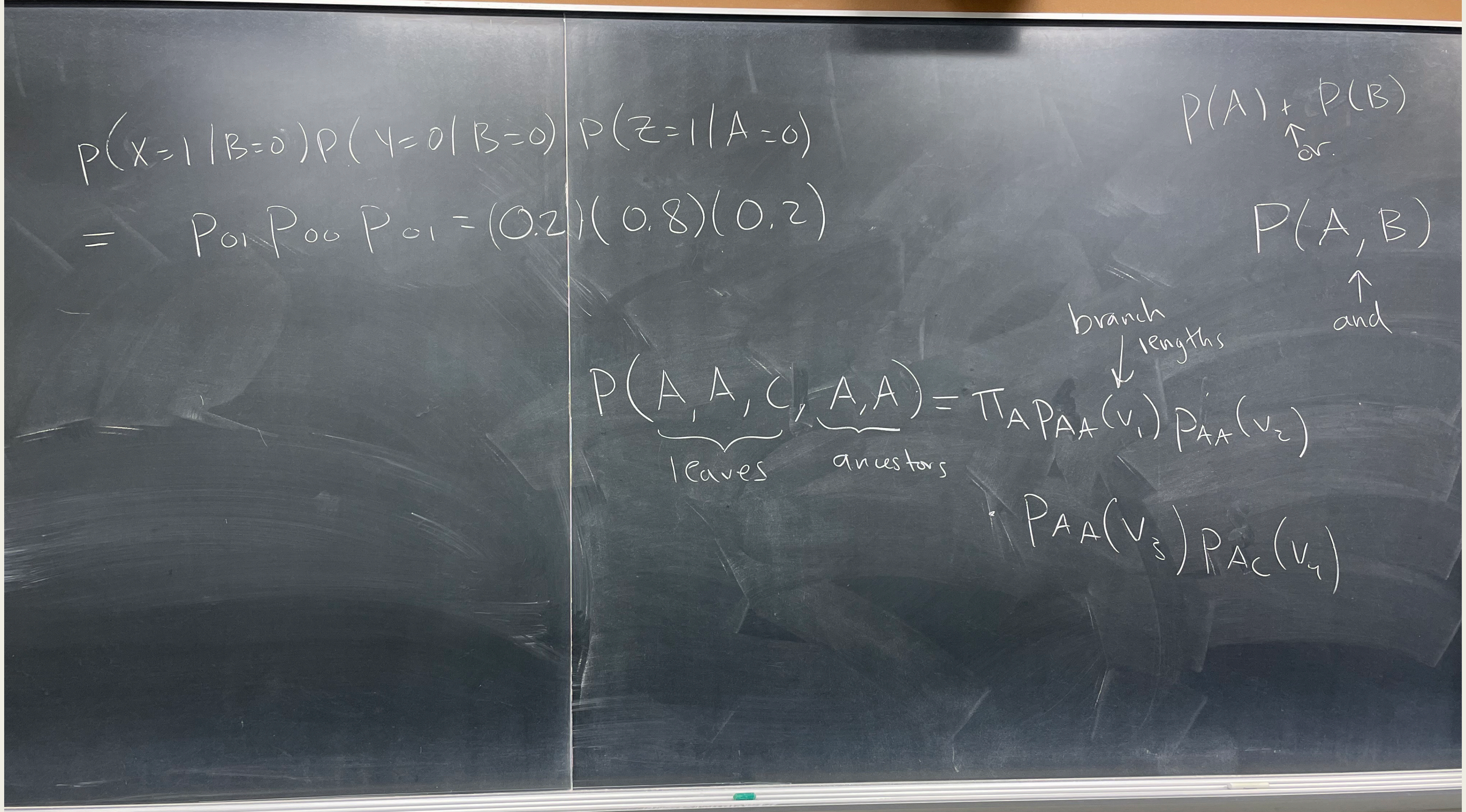
Mutation rate matrix:

	0	1
0	0.8	0.2
1	0.1	0.9

A can be 0,1 and B can be 0,1

Now A and B parameters, so look at $\ell(\mathrm{A},\mathrm{B}) = \mathrm{P}(\mathrm{X},\mathrm{Y},\mathrm{Z}|\mathrm{A},\mathrm{B}) = \mathrm{P}(\mathrm{X}|\mathrm{B})\mathrm{P}(\mathrm{Y}|\mathrm{B})\mathrm{P}(\mathrm{Z}|\mathrm{A})$

$\ell(0,0) = \mathrm{P}(1,0,1|0,0) = p_{01}p_{00}p_{01} = 0.2*0.8*0.2 = 0.032$
$\ell(0,1) = \mathrm{P}(1,0,1|0,1) = p_{11}p_{10}p_{01} = 0.9*0.1*0.2 = 0.018$
$\ell(1,0) = \mathrm{P}(1,0,1|1,0) = p_{01}p_{00}p_{11} = 0.2*0.8*0.9 = 0.144$
$\ell(1,1) = \mathrm{P}(1,0,1|1,1) = p_{11}p_{10}p_{11} = 0.9*0.1*0.9 = 0.081$



$$p(X=1 \mid B=0)\, P(Y=0 \mid B=0)\, P(Z=1 \mid A=0)$$

$$= P_{01} P_{00} P_{01} = (0.2)(0.8)(0.2)$$

$$P(A) \underset{\overset{\uparrow}{\text{or}}}{+} P(B)$$

$$P(A \underset{\overset{\uparrow}{\text{and}}}{,} B)$$

$$P(\underbrace{A, A, C}_{\text{leaves}}, \underbrace{A, A}_{\text{ancestors}}) = \pi_A P_{AA}(\overset{\text{branch lengths}}{\overset{\downarrow}{v_1}}) P_{AA}(v_2)$$

$$P_{AA}(v_3) P_{AC}(v_4)$$

Two issues we need to address

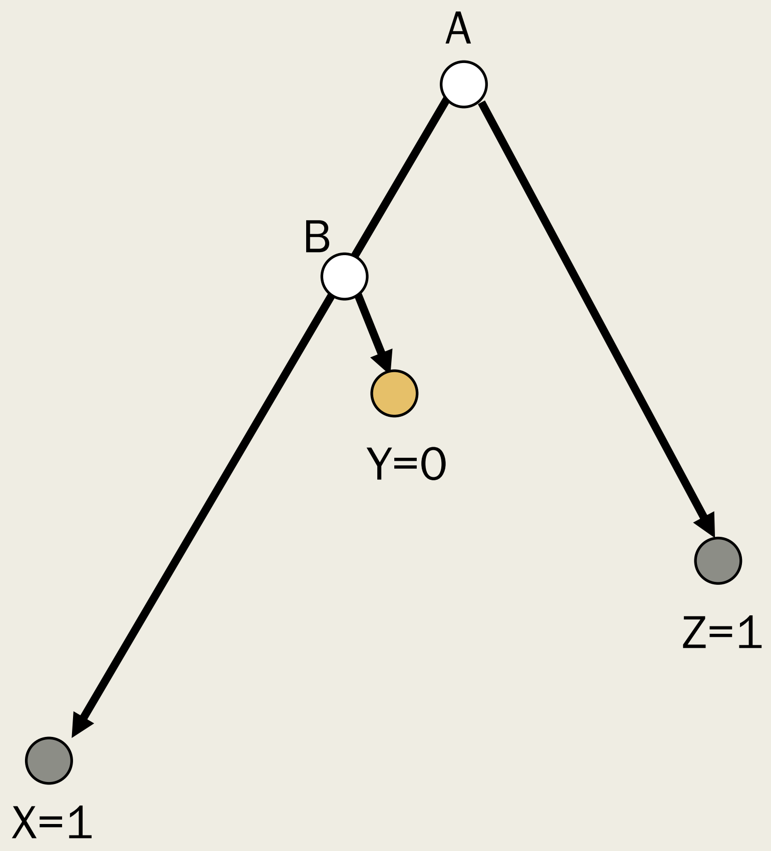
- Probability of changing state should be dependent on the *branch length*

- Solution: think about number of generations for each branch

- Need to work towards the posterior probability, not just likelihood of data given ancestral states

- Solution: Need to integrate over all possibilities for the ancestral states

Incorporating branch length

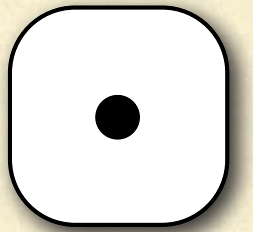


- In general, the probability of a mutation will depend on the branch length
- We can easily calculate this if we assume a constant mutation rate per unit time*
- But now the optimization problem is even harder because we have to optimize over the branch lengths as well as the topology.

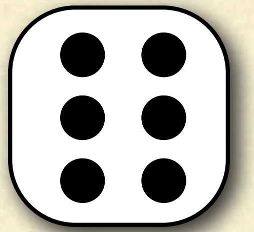
* May not be a good assumption

Probabilities: the AND rule

Rolling 2 dice, what is the probability of seeing (simultaneously) a 1 on the first die and a 6 on the second die?

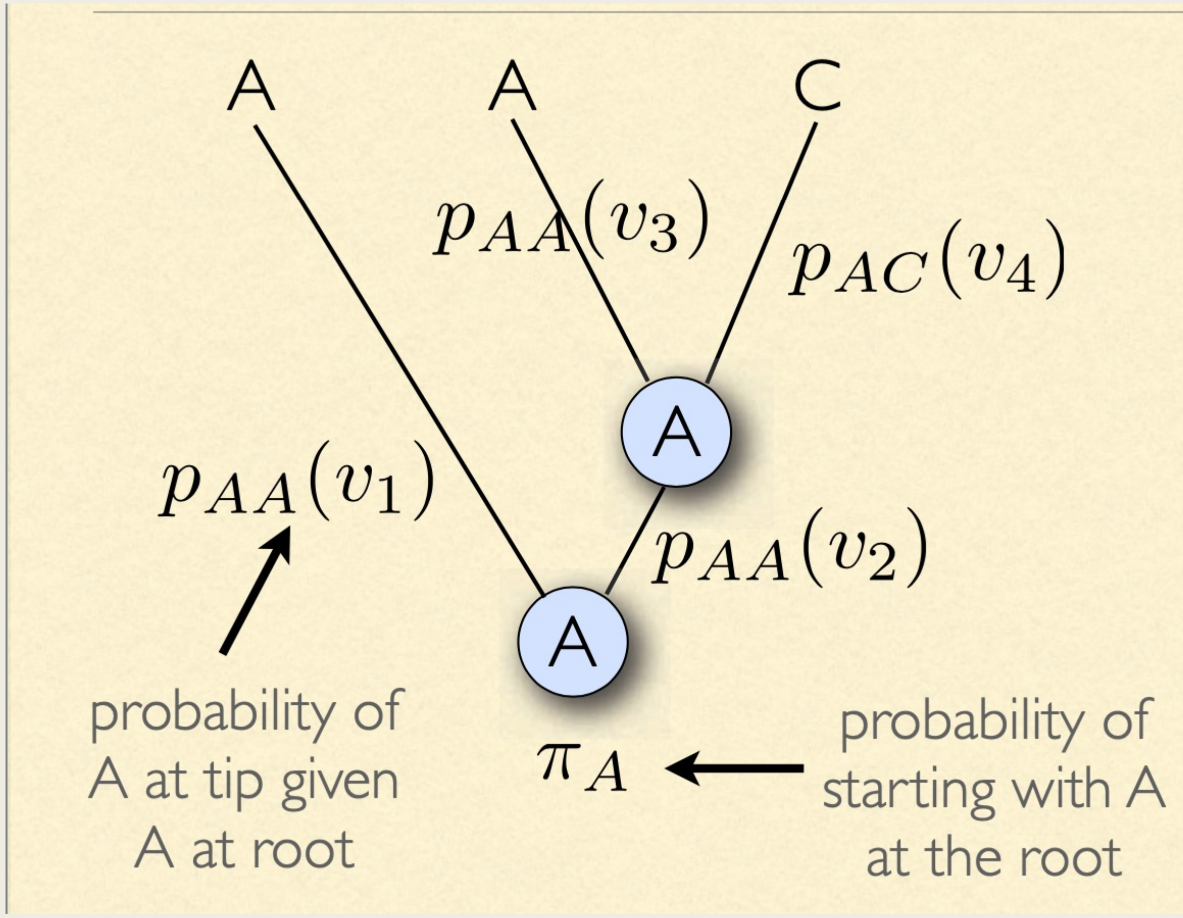


AND



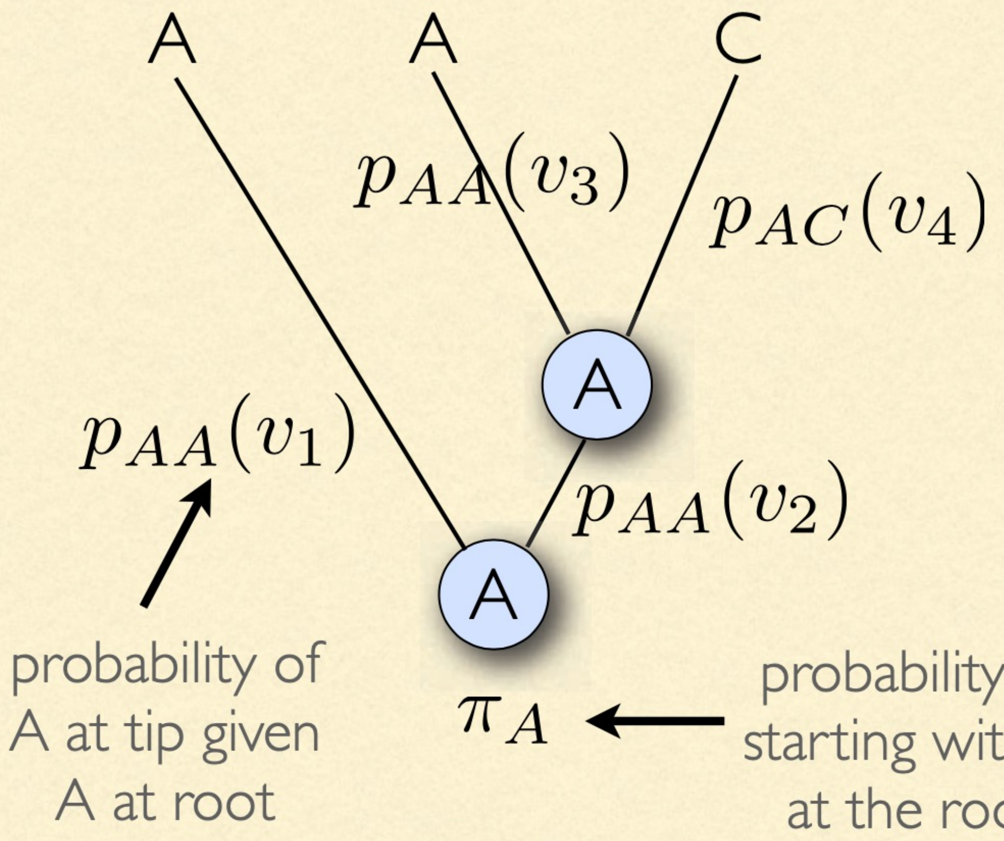
$$(1/6) \times (1/6) = 1/36$$

Probability of: ancestors, leaves given topology, branch lengths


A
A
C
$p_{AA}(v_3)$
$p_{AC}(v_4)$
A
$p_{AA}(v_1)$
$p_{AA}(v_2)$
A
probability of A at tip given A at root
π_A
probability of starting with A at the root

One use of the AND rule in phylogenetics is to combine probabilities associated with individual branches to produce the overall probability of the data for one site.

Probability of: ancestors, leaves given topology, branch lengths

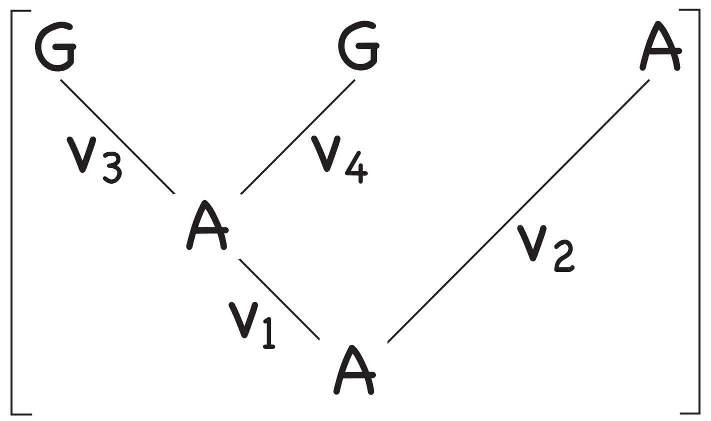

A
A
C
$p_{AA}(v_3)$
$p_{AC}(v_4)$
A
$p_{AA}(v_1)$
$p_{AA}(v_2)$
A
probability of A at tip given A at root
π_A
probability of starting with A at the root

One use of the AND rule in phylogenetics is to combine probabilities associated with individual branches to produce the overall probability of the data for one site.

$$\Pr(A, A, C, A, A) = \pi_A \; p_{AA}(v_1) \; p_{AA}(v_2) \; p_{AA}(v_3) \;\; p_{AC}(v_4)$$

Probability of: ancestors, leaves given topology, branch lengths

Pr



=

$$\pi_A \times p_{AA}(v_1) \times p_{AA}(v_2) \times p_{AG}(v_3) \times p_{AG}(v_4)$$

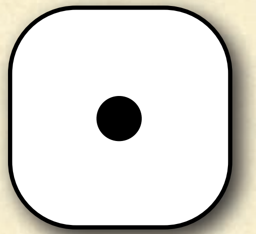
π_i – Stationary frequencies

$p_{ij}(v)$ – Transition probabilities

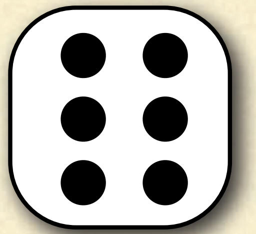
Credit: John Huelsenbeck

Probabilities: the OR rule

Rolling 1 die, what is the probability of seeing either a 1 or a 6?

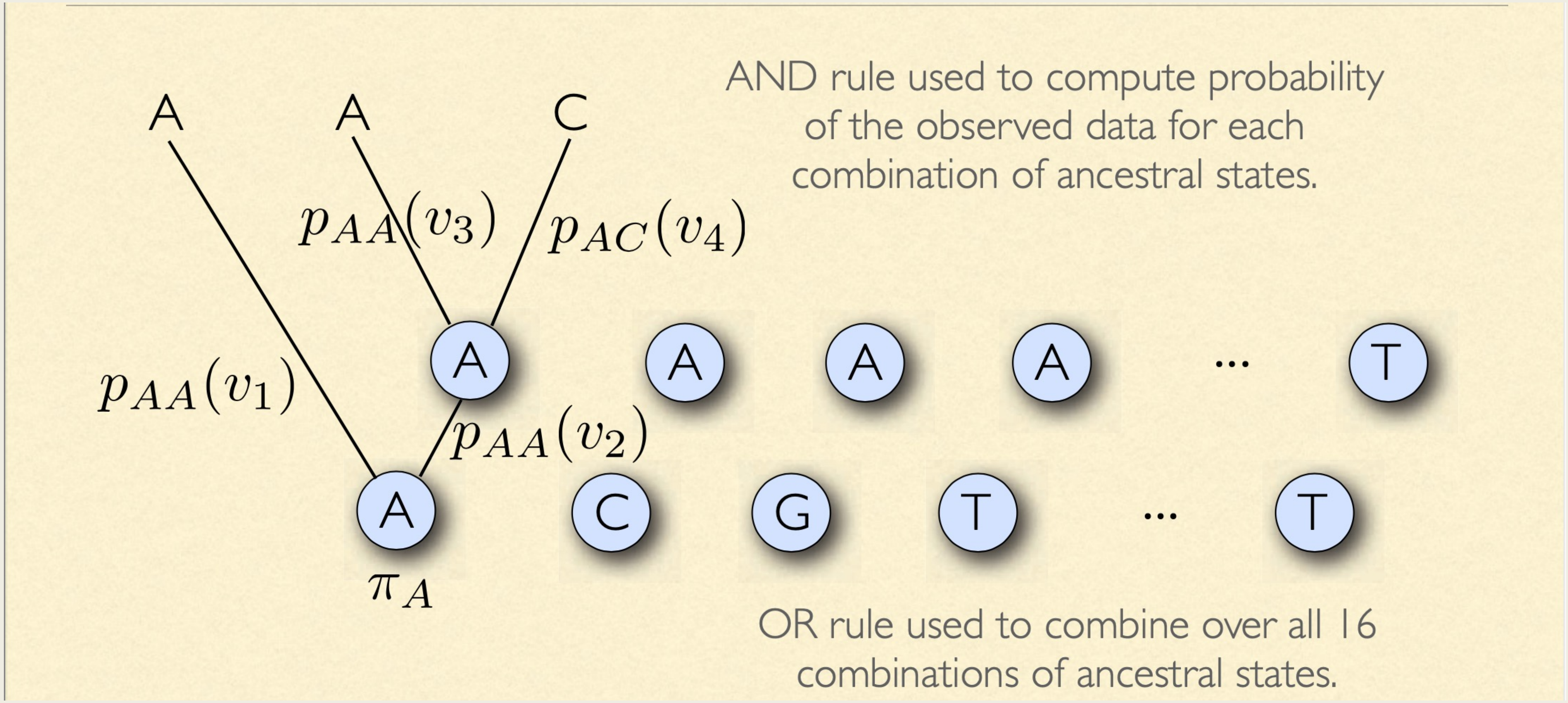


OR



$(1/6) \quad + \quad (1/6) \quad = \quad 1/3$

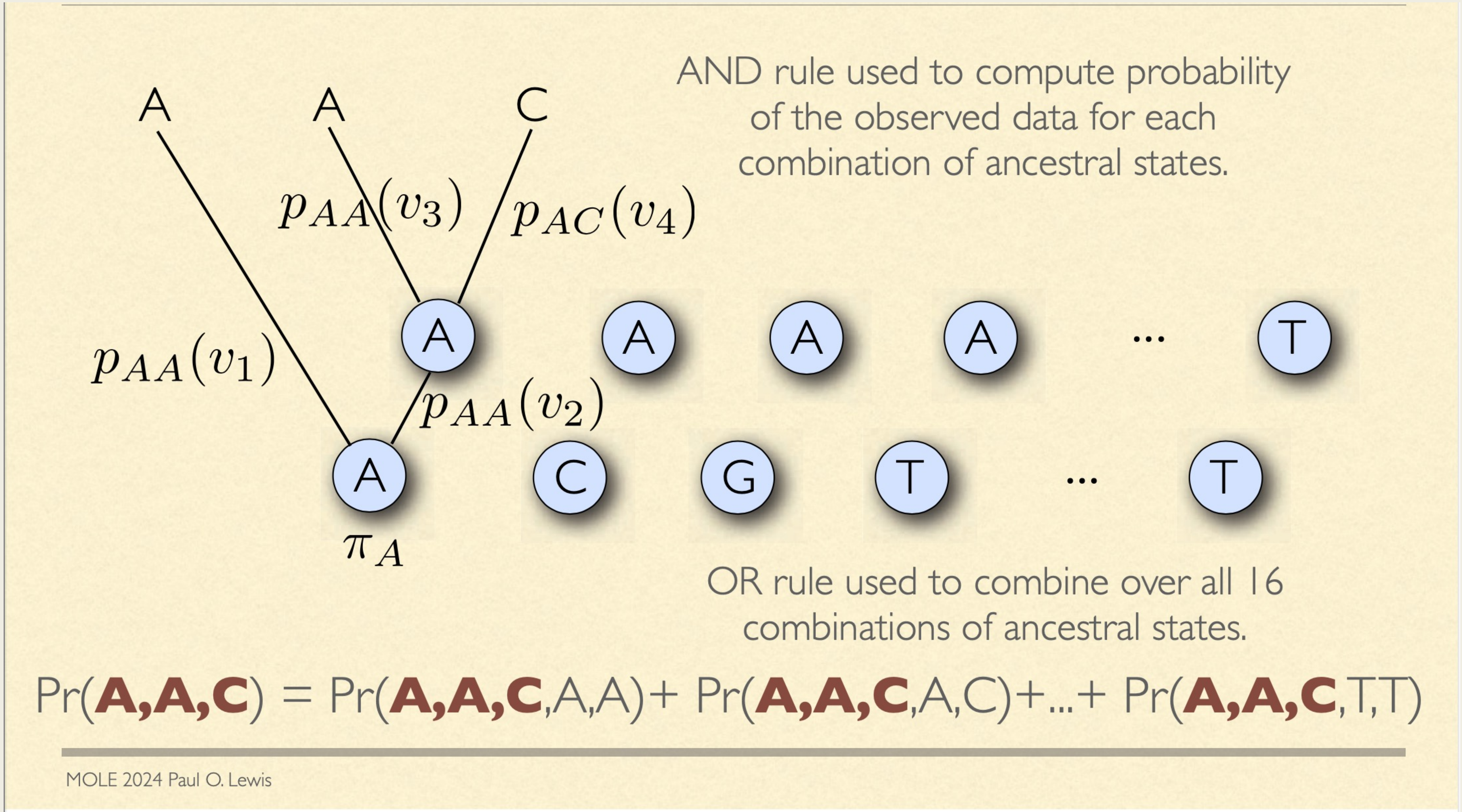
Probability of: leaves given topology, branch lengths



AND rule used to compute probability of the observed data for each combination of ancestral states.

OR rule used to combine over all 16 combinations of ancestral states.

Probability of: leaves given topology, branch lengths



AND rule used to compute probability of the observed data for each combination of ancestral states.

OR rule used to combine over all 16 combinations of ancestral states.

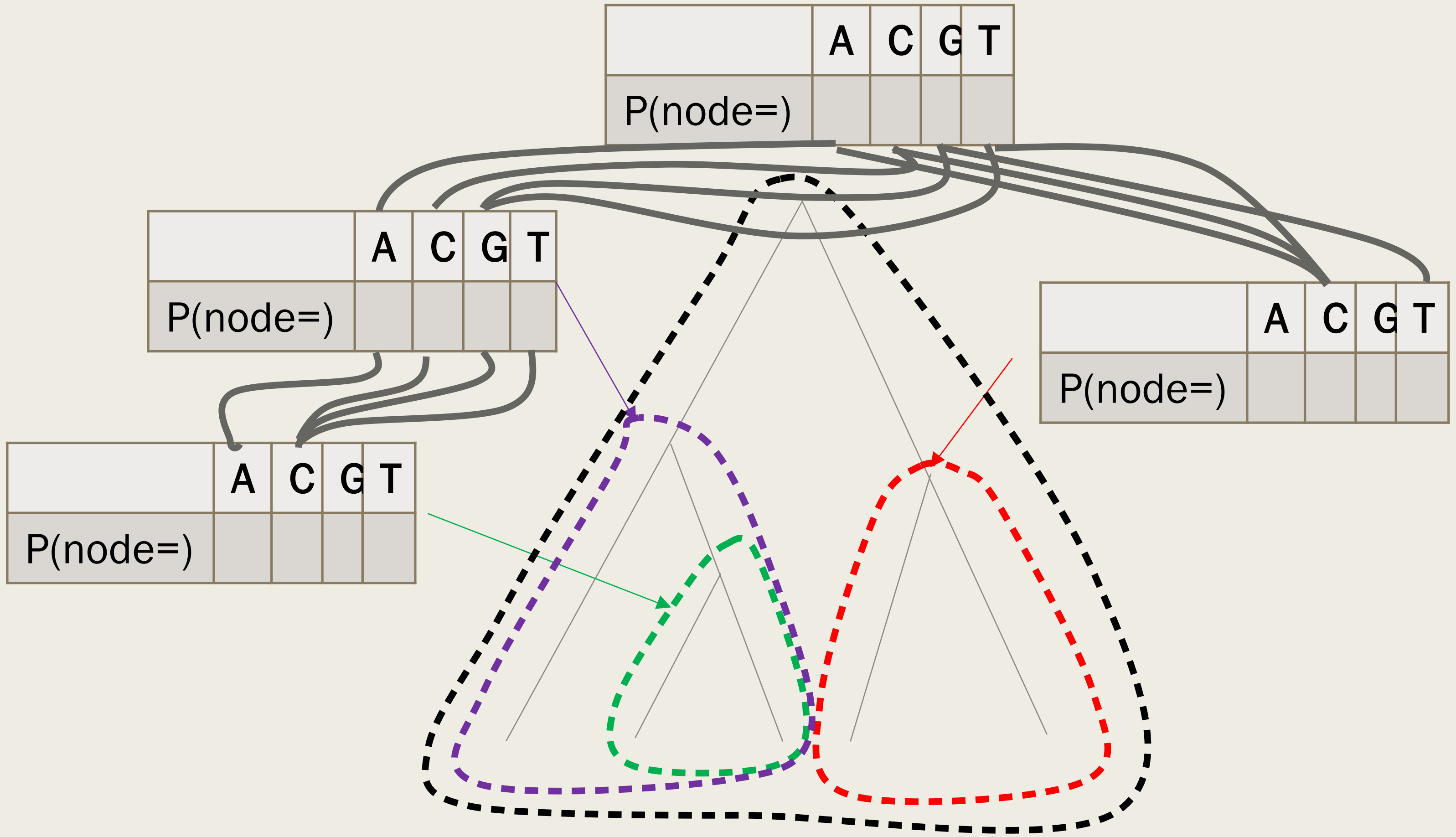
Pr(**A,A,C**) = Pr(**A,A,C**,A,A)+ Pr(**A,A,C**,A,C)+...+ Pr(**A,A,C**,T,T)

MOLE 2024 Paul O. Lewis

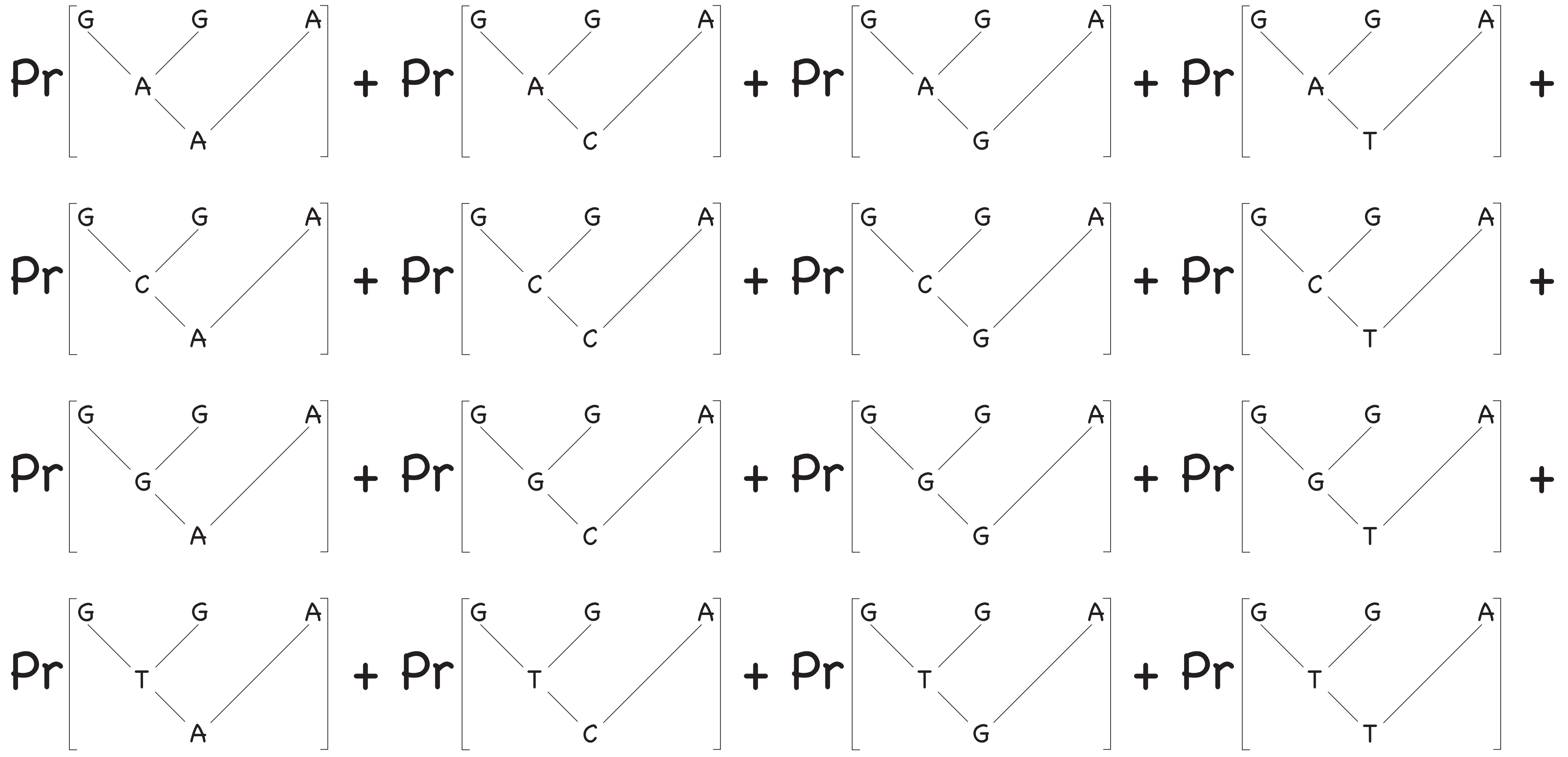
Felsenstein’s Algorithm

Felsenstein’s peeling algorithm

- In general, computing likelihoods is time consuming
- Possible to compute them faster with a dynamic programming algorithm
- This is very similar to Sankoff’s algorithm



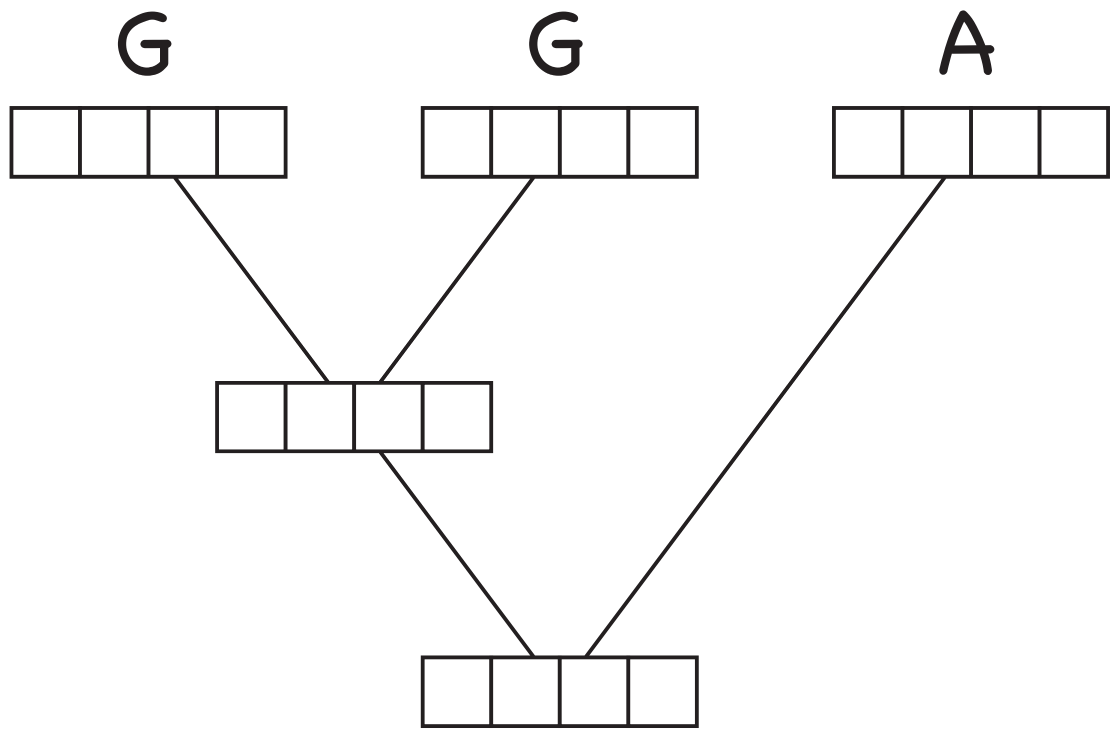
Probability of: leaves given topology, branch lengths



Pr [G G A; A; A] + Pr [G G A; A; C] + Pr [G G A; A; G] + Pr [G G A; A; T] +
Pr [G G A; C; A] + Pr [G G A; C; C] + Pr [G G A; C; G] + Pr [G G A; C; T] +
Pr [G G A; G; A] + Pr [G G A; G; C] + Pr [G G A; G; G] + Pr [G G A; G; T] +
Pr [G G A; T; A] + Pr [G G A; T; C] + Pr [G G A; T; G] + Pr [G G A; T; T]

Credit: John Huelsenbeck

Felsenstein’s peeling algorithm



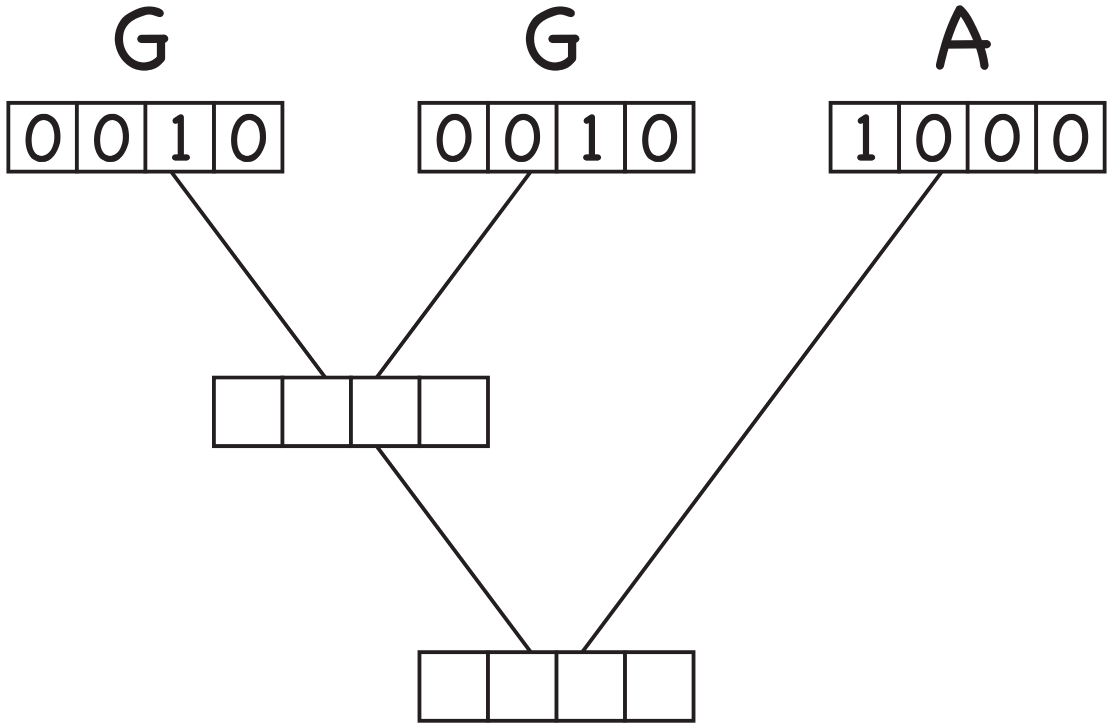
Felsenstein, J. 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. J. Mol. Evol. 17:368–376.

Gallager, R. G. 1962. Low-density parity-check codes. IRE Trans. Inform. Theory 8:21–28.

Gallager, R. G. 1963. Low-density parity-check codes. MIT Press, Cambridge, Mass.

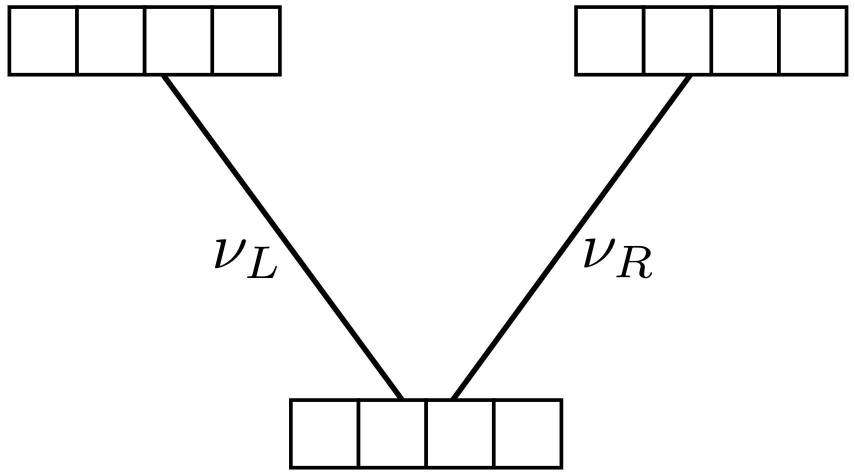
Credit: John Huelsenbeck

Felsenstein’s peeling algorithm



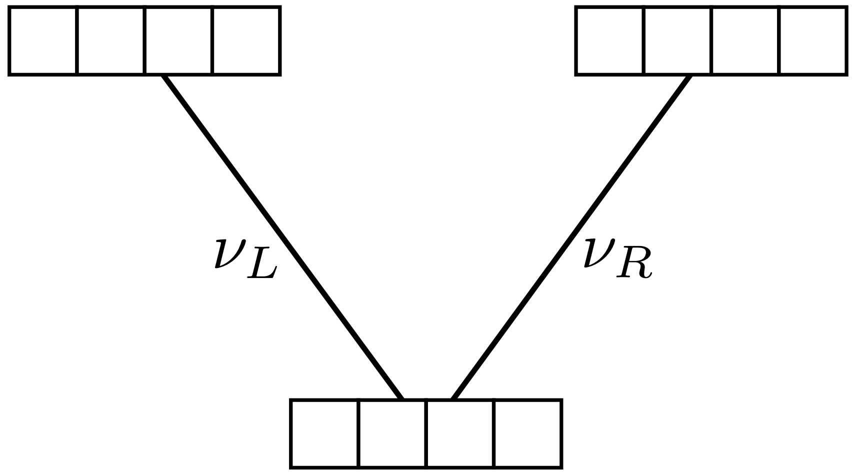
Credit: John Huelsenbeck

Felsenstein’s peeling algorithm



Credit: John Huelsenbeck

Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

$$l_i = \underbrace{\left(\sum_j p_{ij}(\overset{\text{branch lengths}}{v_L})\, l_j^{(L)}\right)}_{\text{left child}} \underbrace{\left(\sum_j p_{ij}(v_R)\, l_j^{(R)}\right)}_{\text{right child}}$$

$i \in \{A, C, G, T\}$

$\boxed{\text{Handout 16}}$ page 1

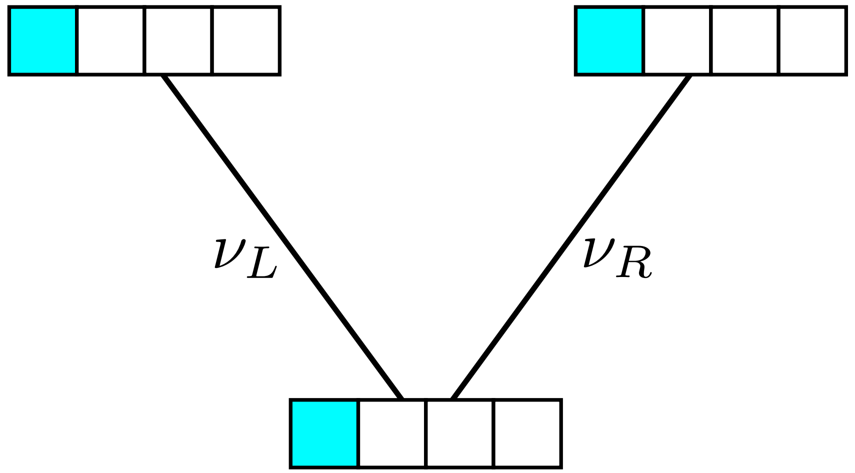
Case 1

$$\begin{tikzcd} \text{Anc} \arrow[d, "\text{2 generations}"'] \\ \text{Der} \end{tikzcd}$$

Case 2

$$\begin{tikzcd} & \text{Anc} \arrow[dl, "\text{2 gen}"'] \arrow[dr, "\text{1 gen}"] & \\ \text{D1} & & \text{D2} \end{tikzcd}$$

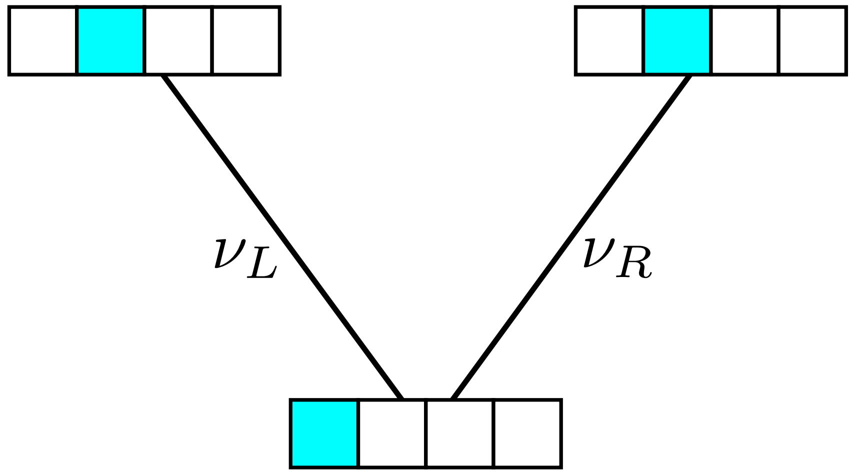
Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

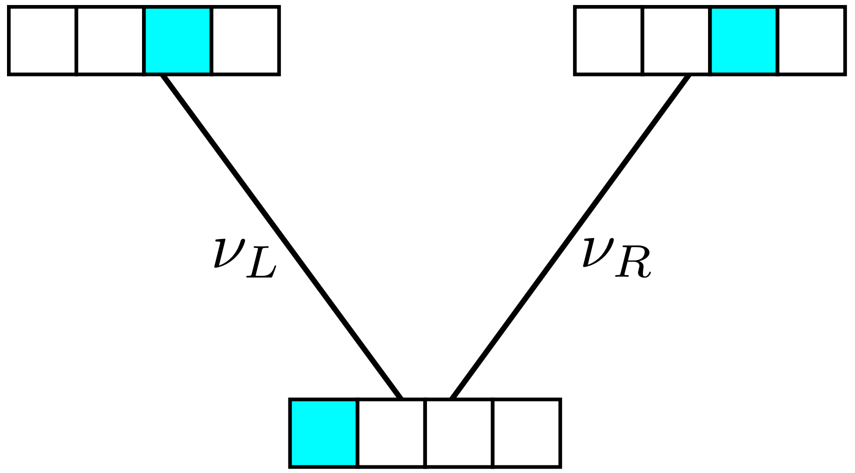
Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

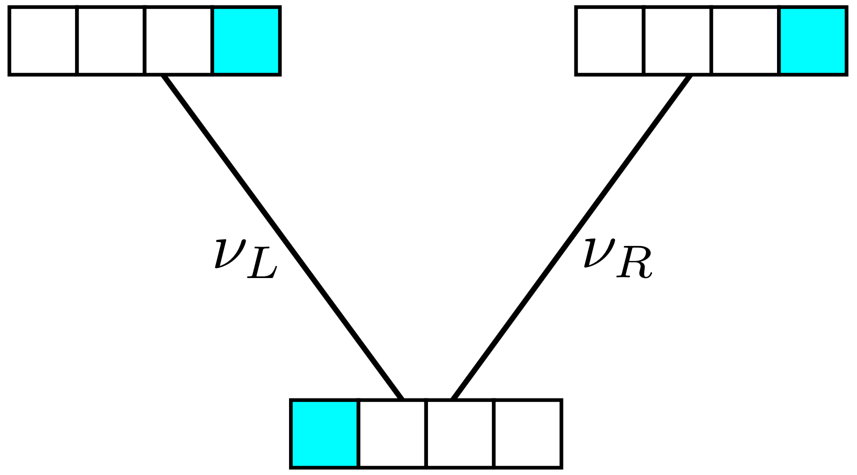
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

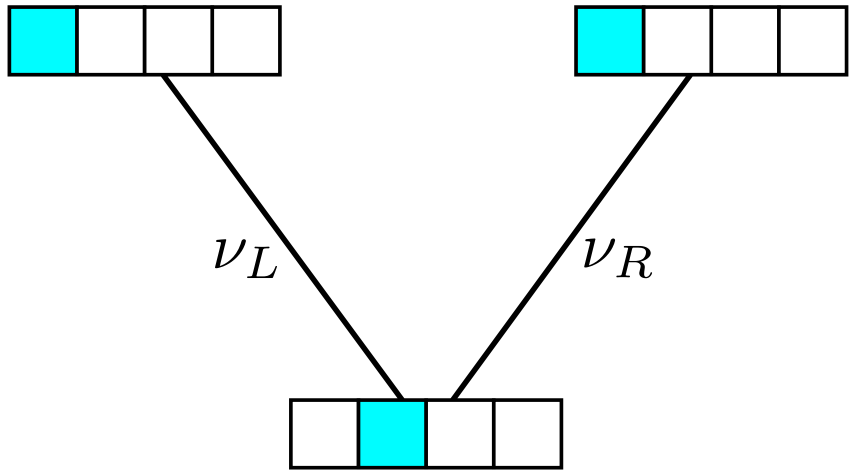
Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

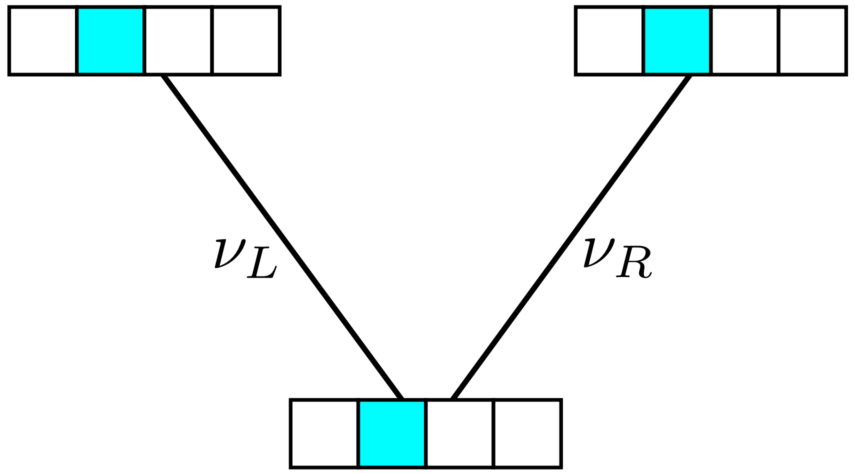
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

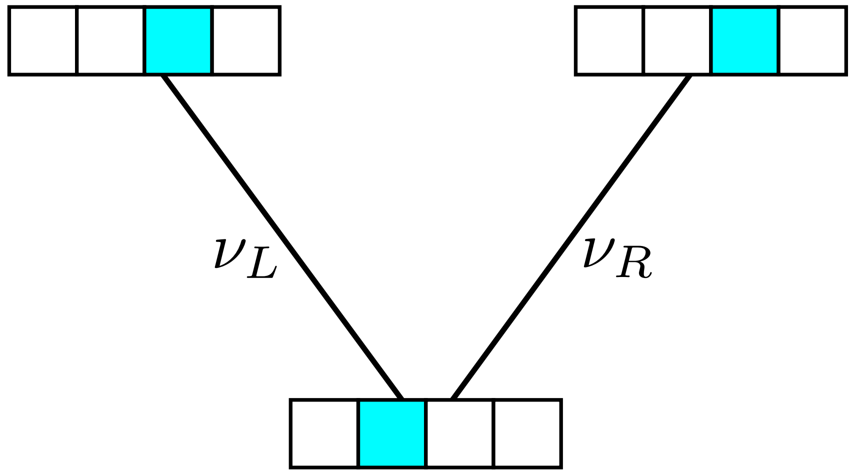
Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

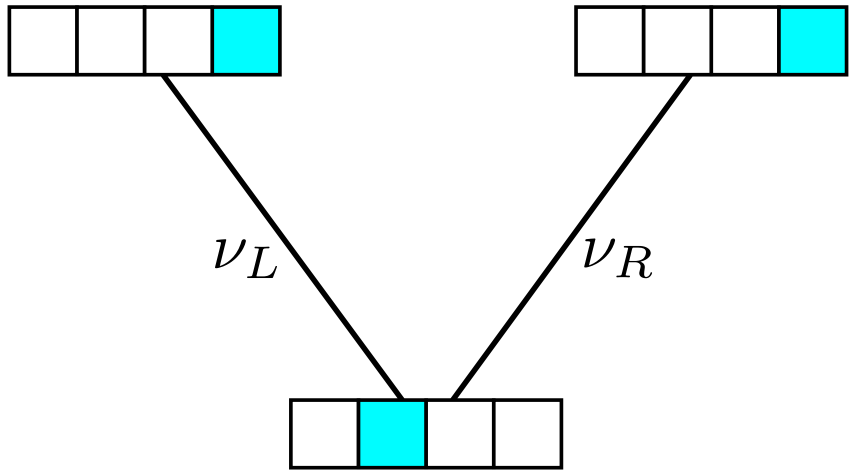
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

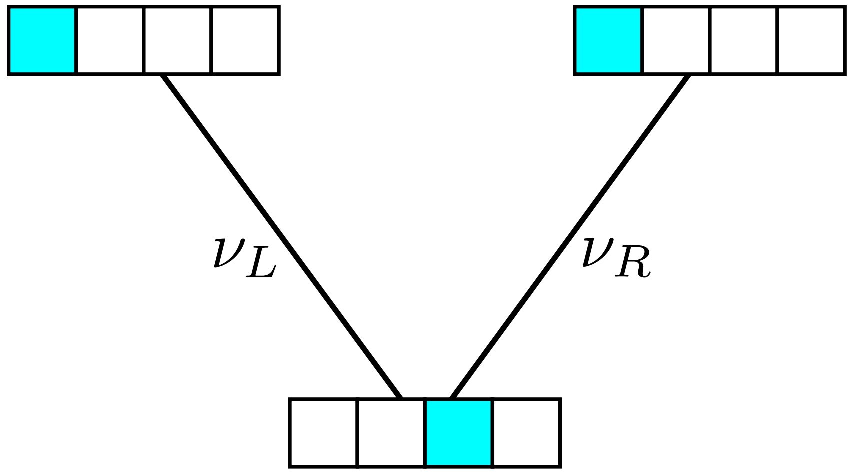
Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

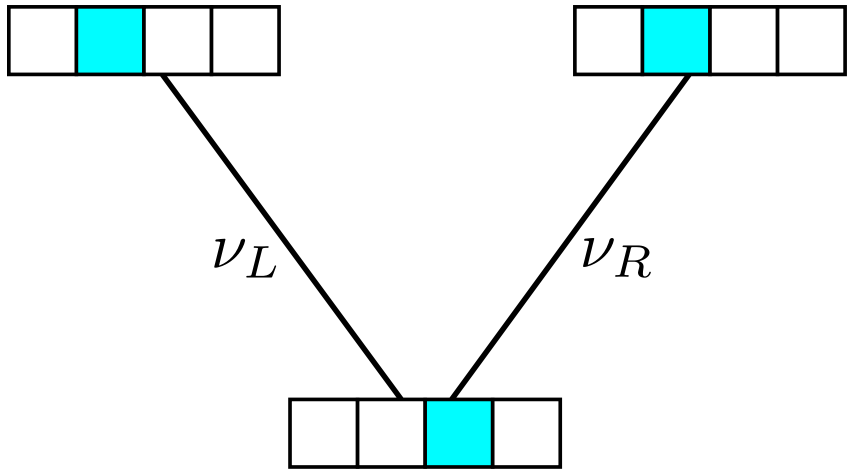
Credit: John Huelsenbeck

Felsenstein’s peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

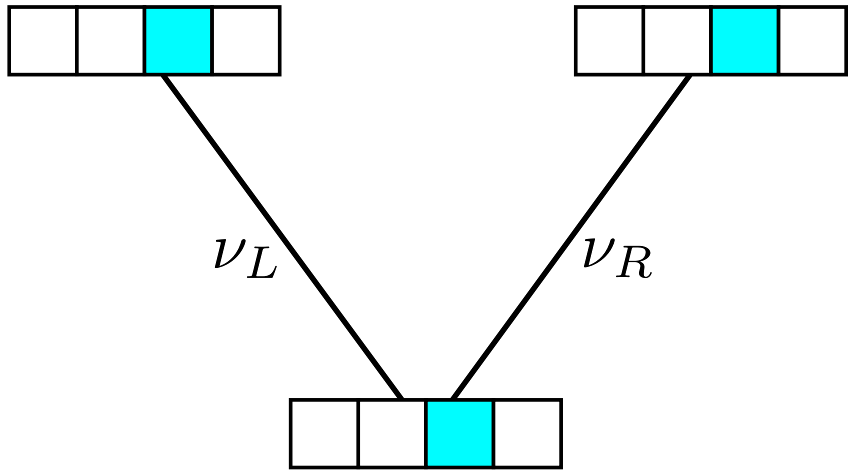
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

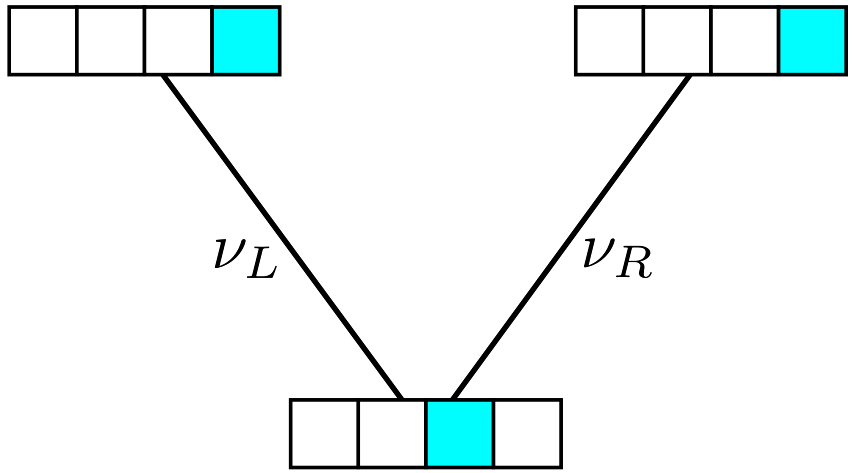
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

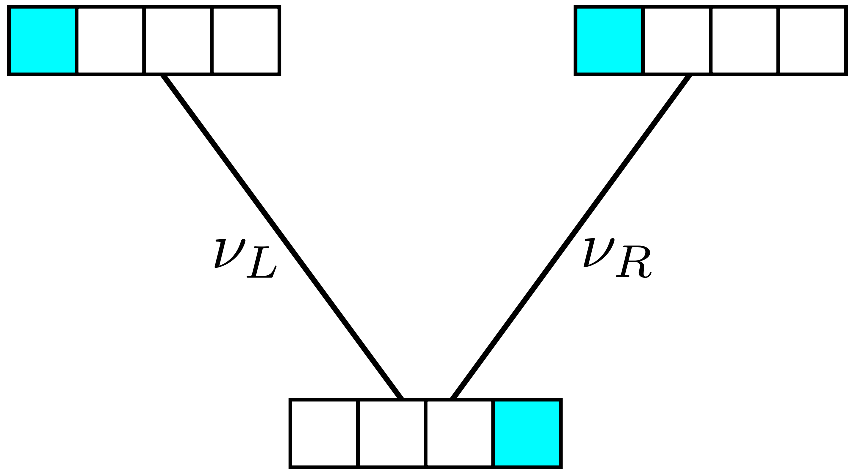
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\,\ell_j^L\right) \times \left(\sum_j p_{ij}(\nu_R)\,\ell_j^R\right)$$

Credit: John Huelsenbeck

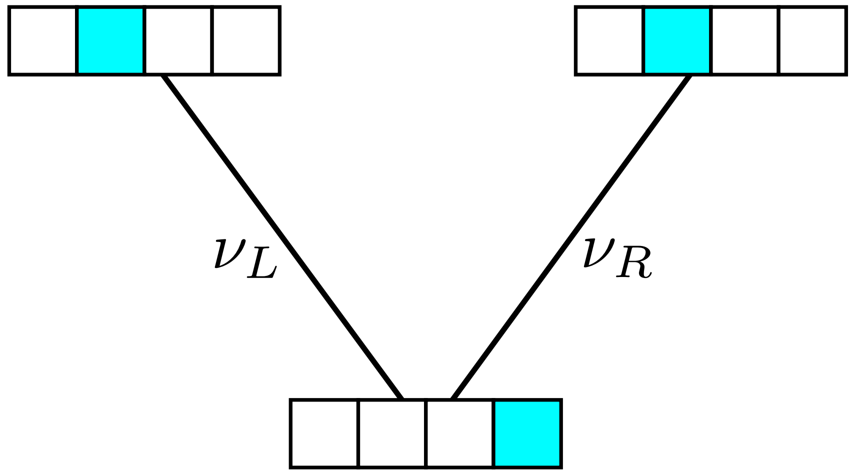
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

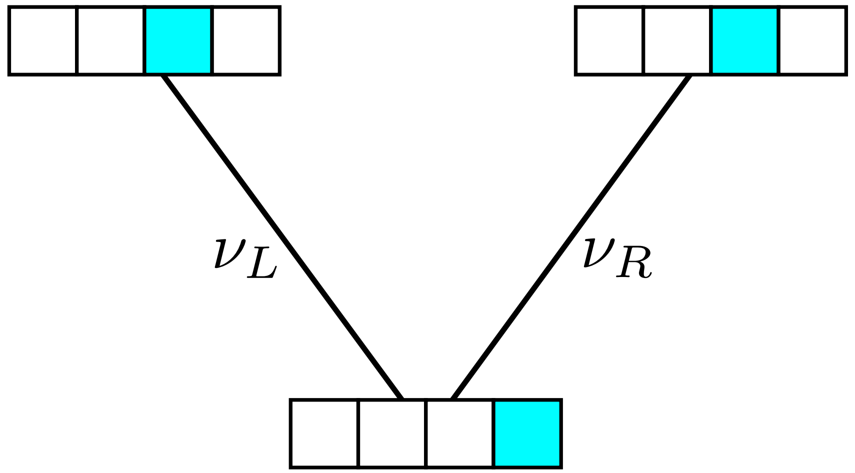
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

Credit: John Huelsenbeck

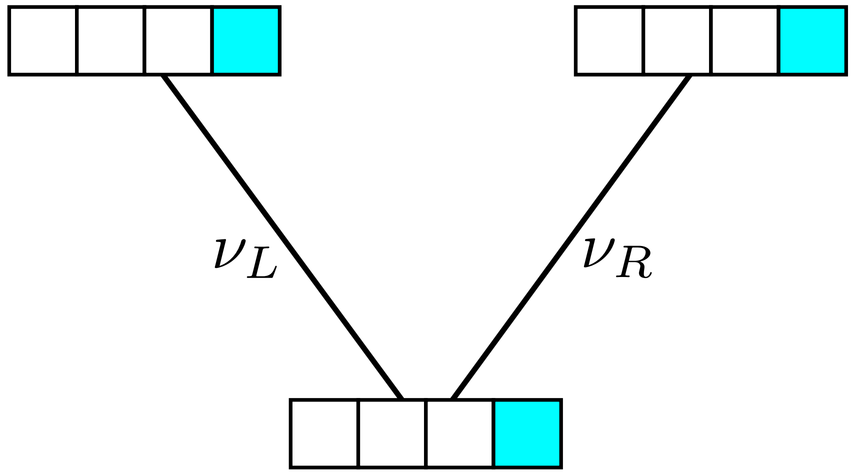
Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

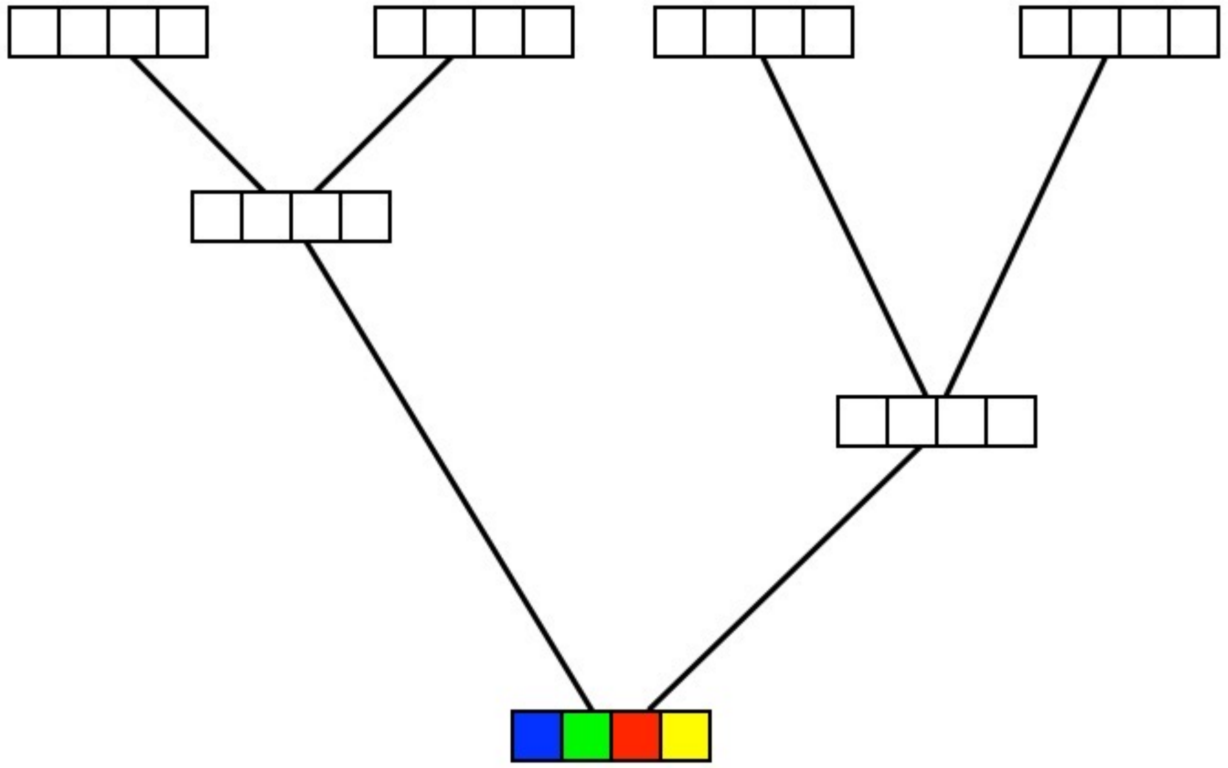
Credit: John Huelsenbeck

Felsenstein's peeling algorithm



$$\ell_i = \left(\sum_j p_{ij}(\nu_L)\, \ell_j^L \right) \times \left(\sum_j p_{ij}(\nu_R)\, \ell_j^R \right)$$

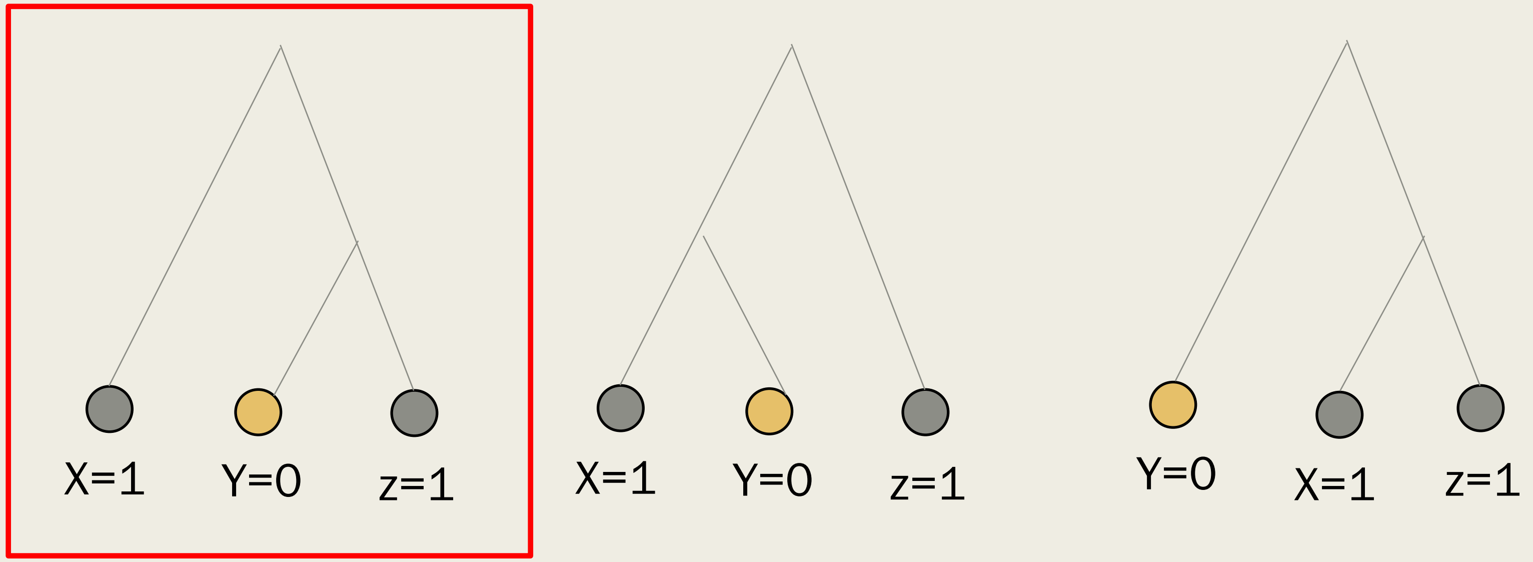
Felsenstein’s peeling algorithm



$$\ell_{\text{Site}} = \pi_A \times \ell_A^{\text{Root}} + \pi_C \times \ell_C^{\text{Root}} + \pi_G \times \ell_G^{\text{Root}} + \pi_T \times \ell_T^{\text{Root}}$$

Credit: John Huelsenbeck

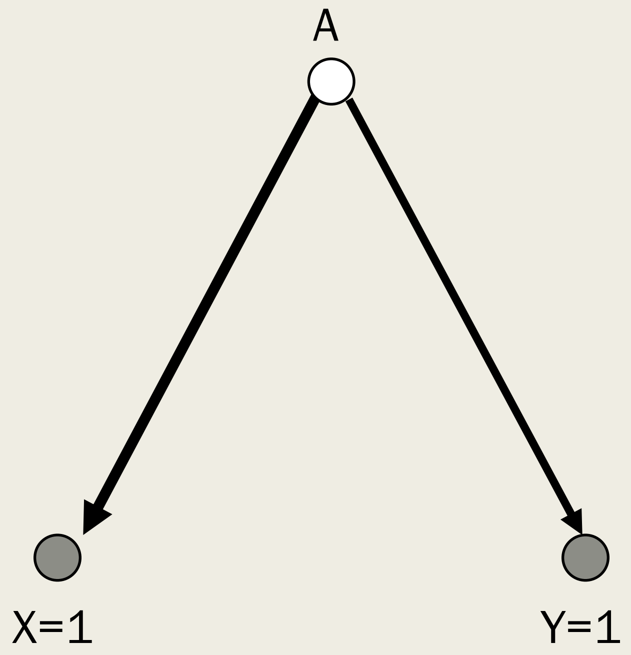
Computing likelihoods of tree topologies



Want to calculate P(tree|data)

To do this for each tree we need to sum over all the possible states of the internal nodes

Computing likelihoods of tree topologies



Simple model: We see two sequences today

What is A?

Mutation rate matrix:

	0	1
0	0.8	0.2
1	0.1	0.9

Either A=0 or A=1, say P(A=0)= π_0, P(A=0)= π_1

Prior probability of the state at the root:
π_0 = probability root is 0 = 0.6
π_1 = probability root is 1 = 0.4

Now the probability is

$$\begin{aligned} P(X,Y) &= P(1,1|A=0)\,\pi_0 + P(1,1|A=1)\,\pi_1 \\ &= 0.04*0.6 + 0.81*0.4 \\ &= 0.348 \end{aligned}$$

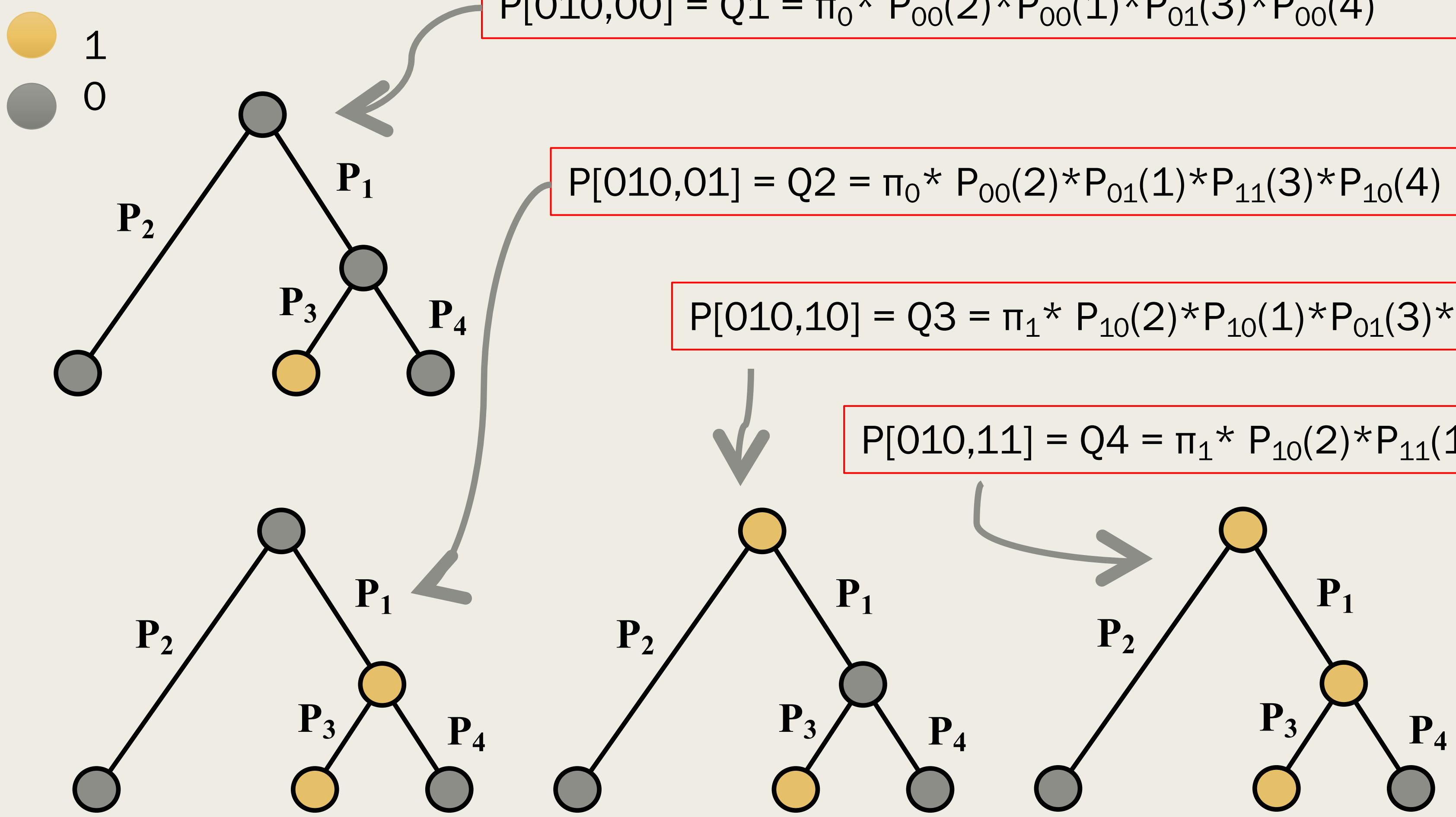
Computing likelihoods of tree topologies

$P[010,00] = Q1 = \pi_0 * P_{00}(2)*P_{00}(1)*P_{01}(3)*P_{00}(4)$

$P[010,01] = Q2 = \pi_0 * P_{00}(2)*P_{01}(1)*P_{11}(3)*P_{10}(4)$

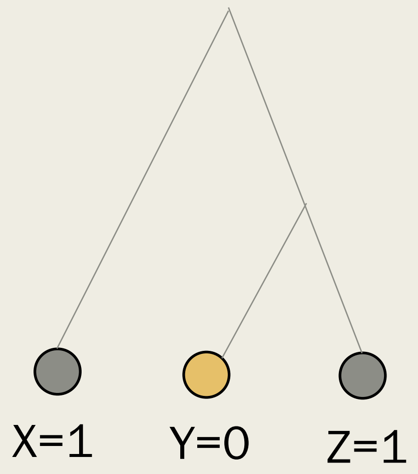
$P[010,10] = Q3 = \pi_1 * P_{10}(2)*P_{10}(1)*P_{01}(3)*P_{00}(4)$

$P[010,11] = Q4 = \pi_1 * P_{10}(2)*P_{11}(1)*P_{11}(3)*P_{10}(4)$

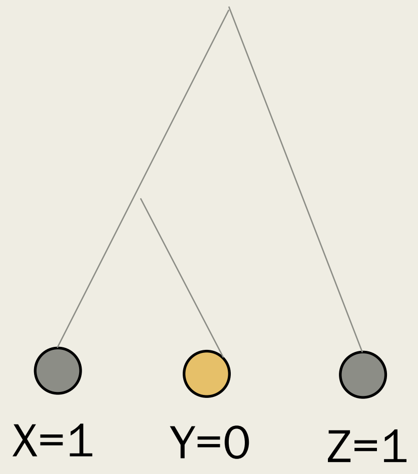


$P(010|\text{topology})=Q1+Q2+Q3+Q4$

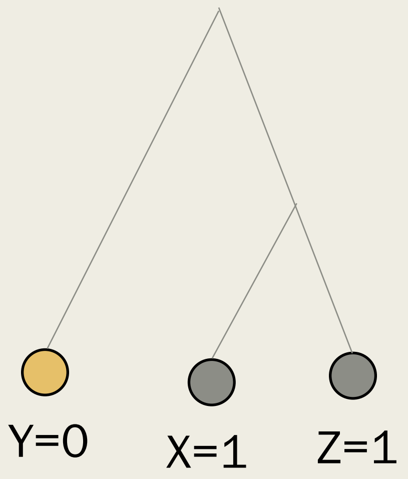
Computing likelihoods of tree topologies



P(010|topology=(X,(Y,Z)))



P(010| topology =(Z,(X,Y)))



P(010| topology =(Y,(X,Z)))

Different mutation models

$$\begin{array}{c|cccc} & A & T & G & C \\ \hline A & Q & p & p & p \\ T & p & Q & p & p \\ C & p & p & Q & p \\ G & p & p & p & Q \end{array}$$

Jukes-Cantor

$$\begin{array}{c|cccc} & A & T & G & C \\ \hline A & Q_1 & p_1 & p_2 & p_1 \\ T & p_1 & Q_2 & p_1 & p_2 \\ C & p_2 & p_1 & Q_3 & p_1 \\ G & p_1 & p_2 & p_1 & Q_4 \end{array}$$

Kimura 2-parameter

$$\begin{array}{c|cccc} & A & T & G & C \\ \hline A & Q_1 & p_1 & p_2 & p_3 \\ T & p_1 & Q_2 & p_4 & p_5 \\ C & p_2 & p_4 & Q_3 & p_6 \\ G & p_3 & p_5 & p_6 & Q_4 \end{array}$$

Tamura-Nei

$$\begin{array}{c|cccc} & A & T & G & C \\ \hline A & Q_1 & p_1 & p_2 & p_3 \\ T & p_4 & Q_2 & p_5 & p_6 \\ C & p_7 & p_8 & Q_3 & p_9 \\ G & p_{10} & p_{11} & p_{12} & Q_4 \end{array}$$

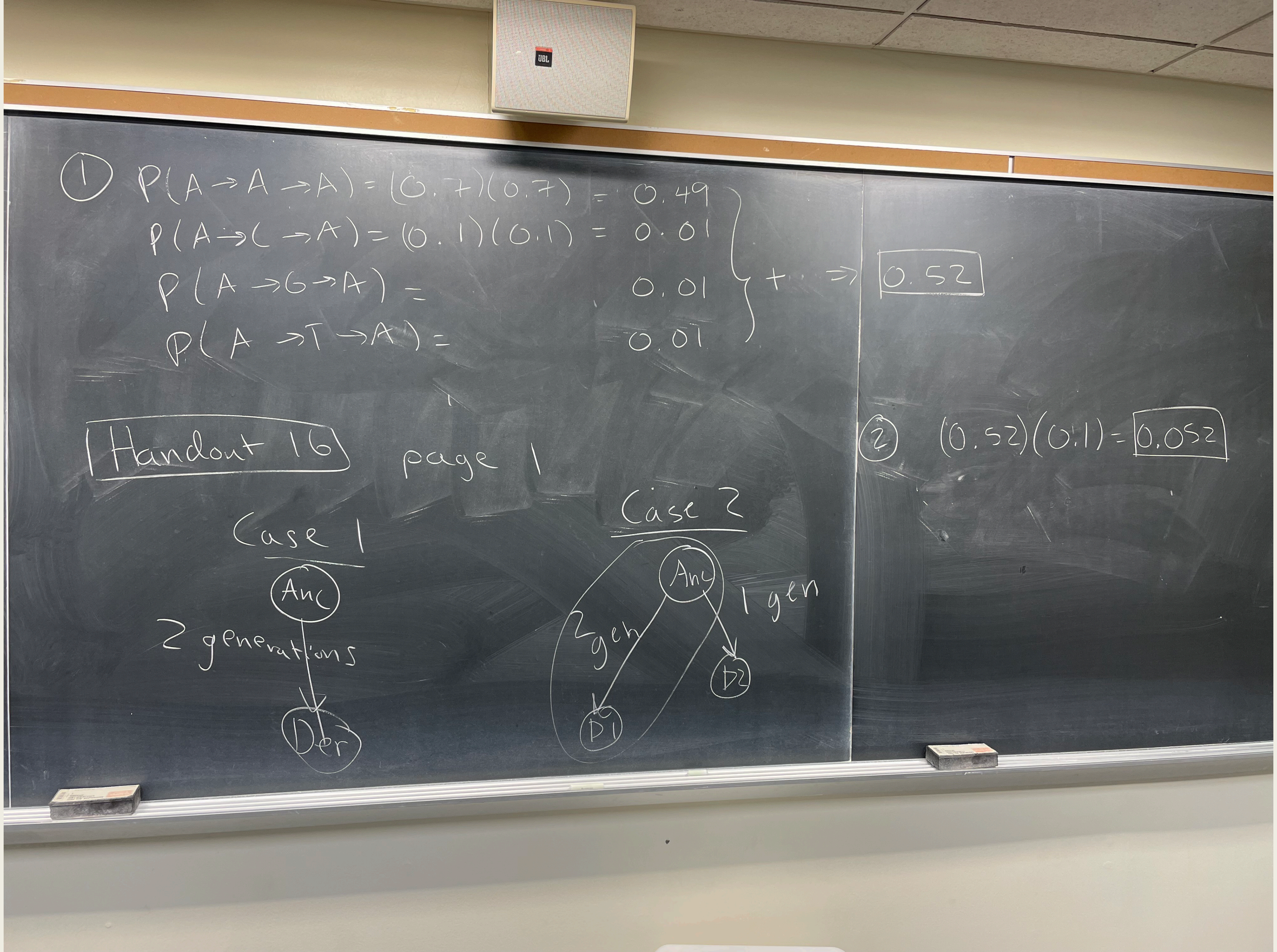
General 12-parameter

https://plewis.github.io/applets/jc-transition-probabilities/

Finding the maximum likelihood tree

- We can compute the likelihood of a tree, given the data (just like computing the parsimony score).
- But to find the best tree, we still have to search through the space of all trees (which there are exponentially many and the problem is NP-hard etc…)
- We can use heuristic methods etc, just as for parsimony.
- But in practice this is only practical for relatively small problems
- So what does maximum likelihood get us? It's interpretable, it's easily extended, and it allows us to use all the statistical approaches that have been developed around likelihood methods.
- Importantly, gives us a measure of uncertainty.

Tree likelihood worksheet



① $P(A \to A \to A) = (0.7)(0.7) = 0.49$
$P(A \to C \to A) = (0.1)(0.1) = 0.01$
$P(A \to G \to A) = \quad 0.01$
$P(A \to T \to A) = \quad 0.01$

$\left.\right\} + \Rightarrow \boxed{0.52}$

② $(0.52)(0.1) = \boxed{0.052}$

$\boxed{\text{Handout 16}}$ page 1

Case 1

$$\begin{tikzcd}
\text{Anc} \arrow[d, "\text{2 generations}"'] \\
\text{Der}
\end{tikzcd}$$

Case 2

$$\begin{tikzcd}
 & \text{Anc} \arrow[dl, "\text{2 gen}"'] \arrow[d, "\text{1 gen}"] \\
\text{D1} & \text{D2}
\end{tikzcd}$$

Ways forward:

1) bootstrap, 2) MCMC

Bootstrap (sampling sites with replacement)

The Bootstrap

In an 18th century story by Rudolph Erich Raspe, Baron Munchausen falls to the bottom of a deep lake.

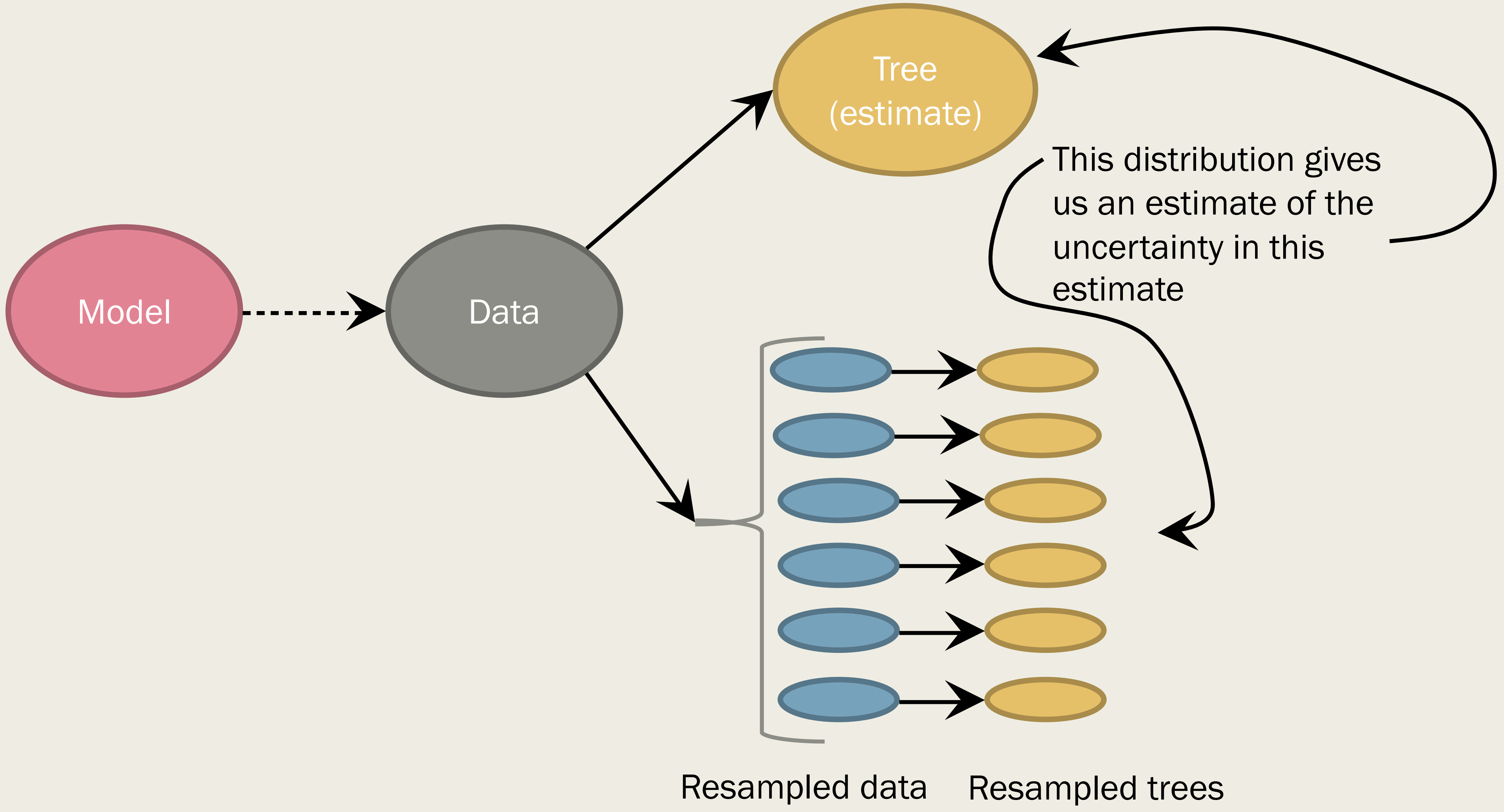
About to drown, he has the idea to lift himself up by pulling on his bootstraps

(In the original German version, he pulls himself up by his hair, left).

Obviously impossible, this story gave its name to a statistical technique (Efron, 1795) that seems magical, in the sense that you can get something (estimates of uncertainty) for nothing!

In general, the bootstrap is an incredibly useful statistical technique – perhaps one of the most useful in all of modern statistics.

Bootstrapping


Tree
(estimate)
This distribution gives us an estimate of the uncertainty in this estimate
Model
Data
Resampled data
Resampled trees

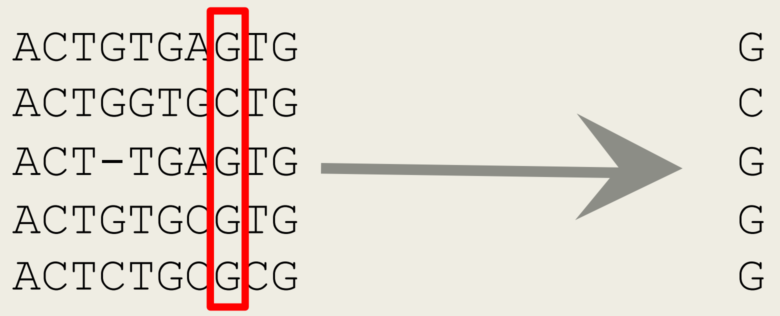
The bootstrap: Resampling

- The key point is that as long as we can resample our data (which we can always do).
- And calculate the thing we want to estimate (which we can almost always do).
- We can bootstrap anything, and get a sense of how good our estimate is.
- We do not need to make any assumptions about the underlying distribution. For example, to apply the central limit theorem.

Resampling molecular data

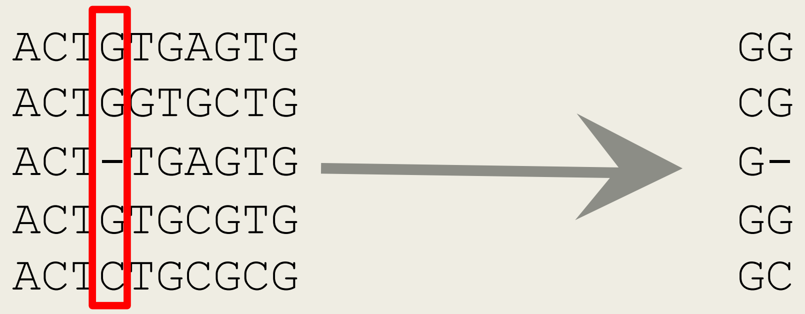
```
ACTGTGAGTG
ACTGGTGCTG
ACT-TGAGTG
ACTGTGCGTG
ACTCTGCGCG
```

Resampling molecular data



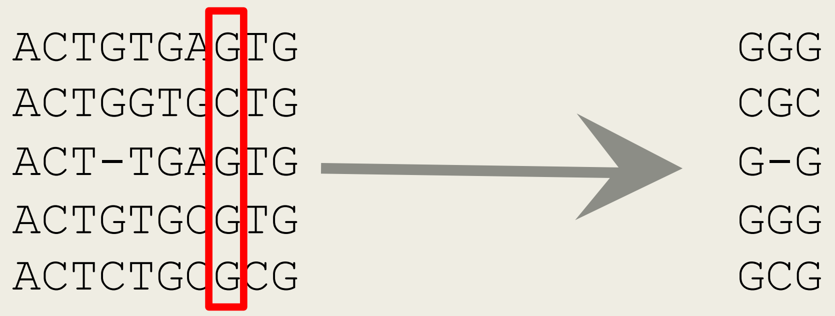
```
ACTGTGAGTG        G
ACTGGTGCTG        C
ACT-TGAGTG        G
ACTGTGCGTG        G
ACTCTGCGCG        G
```

Resampling molecular data



```
ACTGTGAGTG          GG
ACTGGTGCTG          CG
ACT-TGAGTG          G-
ACTGTGCGTG          GG
ACTCTGCGCG          GC
```

Resampling molecular data



```
ACTGTGAGTG        GGG
ACTGGTGCTG        CGC
ACT-TGAGTG        G-G
ACTGTGCGTG        GGG
ACTCTGCGCG        GCG
```

Resampling molecular data

```
ACTGTGAGTG
ACTGGTGCTG
ACT-TGAGTG
ACTGTGCGTG
ACTCTGCGCG
```

$\longrightarrow$

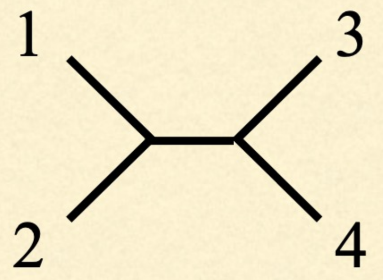
```
GGGAAGCAGG
CGCAATCGCG
G-GAAGCAG-
GGGAAGCCGG
GCGAAGCCGC
```

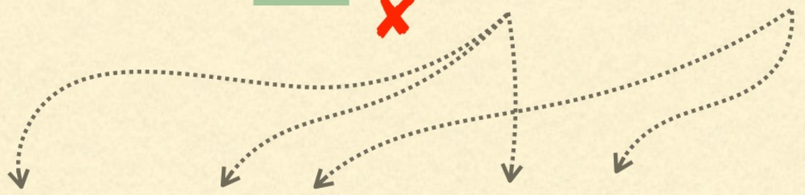
Bootstrap support

sites sampled with replacement

original data

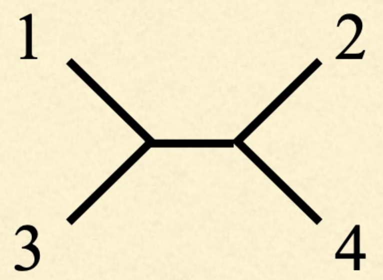
	1	2	3	4	5	6	7	8
1	A	G	G	C	G	T	A	C
2	A	A	G	C	G	T	A	T
3	A	G	T	C	A	C	G	G
4	A	A	T	C	G	C	G	G
	✘					✘	✘	



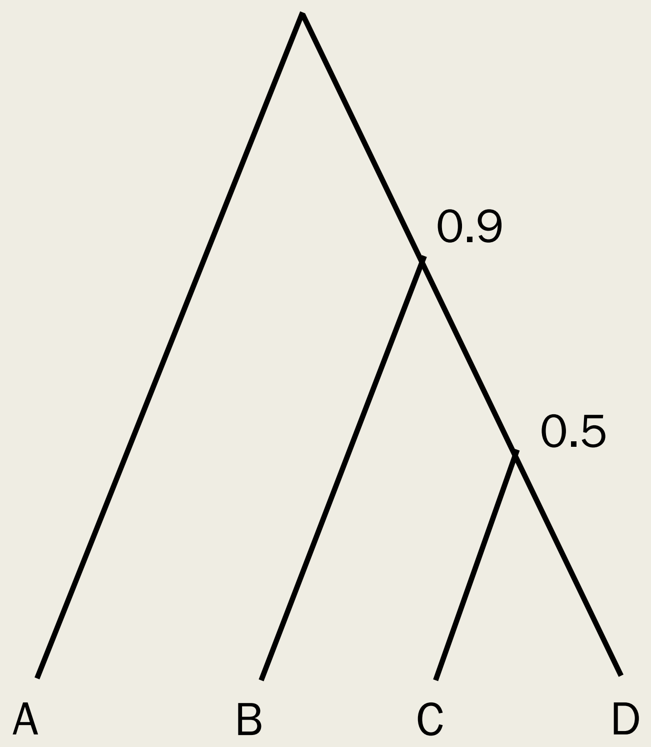


bootstrap replicate

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



How to read bootstrap values for trees

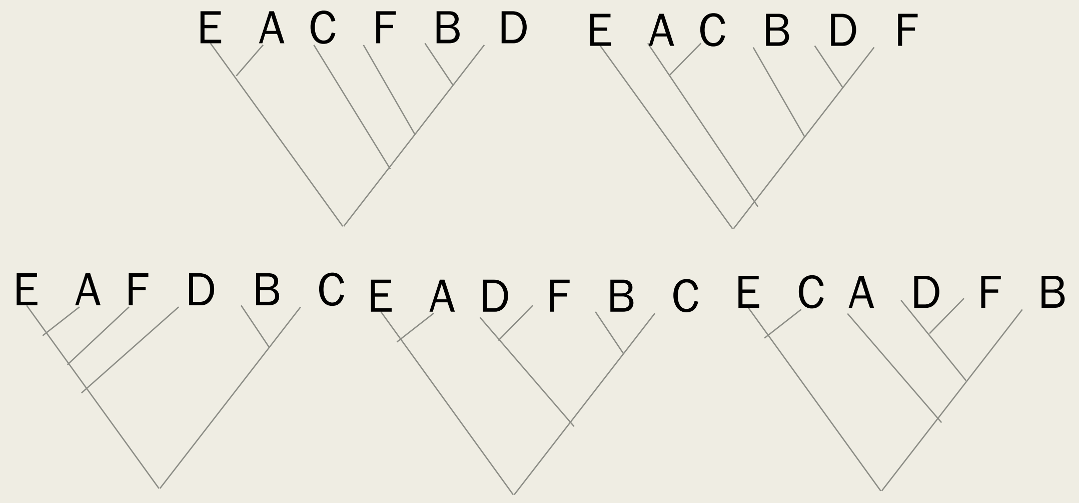


{B,C,D} clade appeared in 90% of bootstrap trees

{C,D} appeared in 50%

So the interpretation is that A is probably an outgroup, but we cannot identify the relationship between B,C and D.

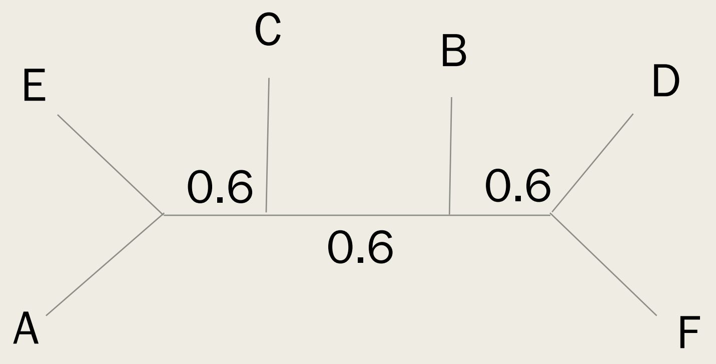
Majority rule consensus trees



How many times each partition is found:

AE | BCDF 3
ACE | BDF 3
ACEF | BD 1
AC | BDEF 1
AEF | BCD 1
ADEF | BC 2
ABDF | EC 1
ABCE | DF 3

Majority consensus tree



Example: Joe Felsenstein

Bootstrap summary

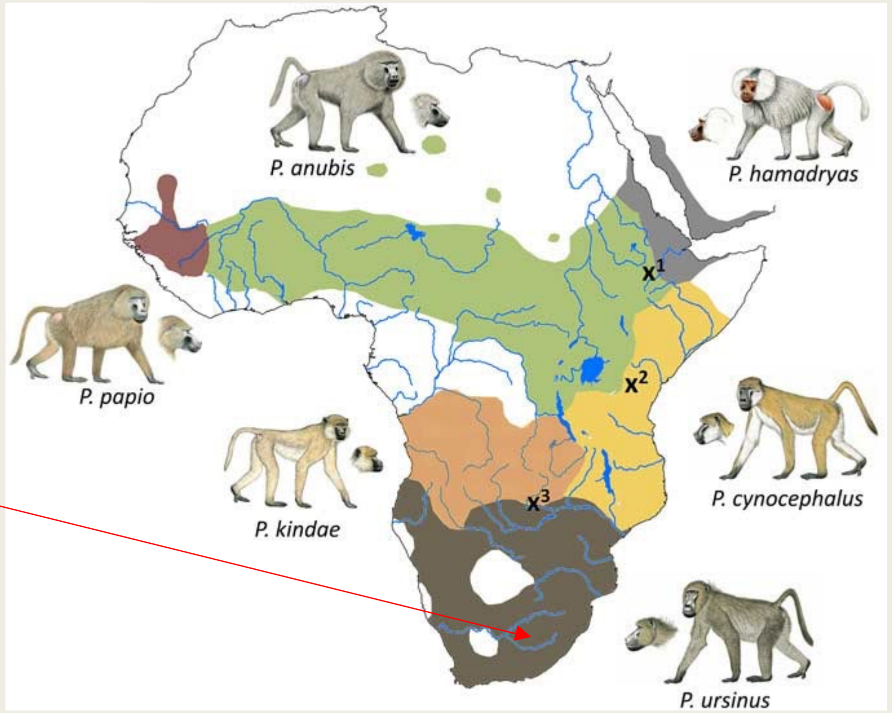
1. Resample columns of character matrix

2. Build tree using resampled character matrix (i.e. recompute distance matrix and run UPGMA/NJ)

3. Compare and report summary of all the resampled trees; e.g. support values, likelihood, consensus trees etc…

Bootstrapping worksheet

Next time!



“An Ancient Baboon Genome Demonstrates Long-Term Population Continuity in Southern Africa”, *GBE* (2020)