Phylogenetic reconstruction: criteria

Bastien Boussau
Bastien.boussau@univ-lyon1.fr
@bastounette

LBBE
BIOMETRIE ET BIOLOGIE EVOLUTIVE

cnrs

UCBL Lyon 1

Phylogenetic inference

How to find the best tree given my data?

- **Need for a criterion/score**

- **Need for an algorithm to find/construct the tree**

Phylogenetic inference

How to find the best tree given my data?

- **Need for a criterion/score**
  - Maximum Parsimony
  - Minimum Evolution or least squares (distance methods)
  - Maximum Likelihood ~ P(D|M)
  - Posterior Probability P(M|D)
- **Need for an algorithm to find/construct the tree**

Phylogenetic inference

How to find the best tree given my data?

- **Need for a criterion/score**
  - Maximum Parsimony
  - Minimum Evolution or least squares (distance methods)
  - Maximum Likelihood ~ P(D|M)
  - Posterior Probability P(M|D)
- **Need for an algorithm to find/construct the tree**

Phylogenetic inference

How to find the best tree given my data?

- **Need for a criterion/score**
  - Maximum Parsimony
  - Minimum Evolution or least squares (distance methods)
  - Maximum Likelihood ~ P(D|M)
  - Posterior Probability P(M|D)
- **Need for an algorithm to find/construct the tree**
  - e.g.: try several topologies, (choose some branch lengths,) score the topologies, choose the one that has the best score

Plan: Criteria for evaluating phylogenies

- Criteria for evaluating phylogenetic trees:
  - Parsimony
  - Distance methods
  - Maximum Likelihood
  - Posterior probability (Bayesian approach)
- Conventions:
  - We’re dealing with aligned sequence data
  - gaps are not taken into account

Parsimony

- “The principle that the most acceptable explanation of an occurrence, phenomenon, or event is the simplest, involving the fewest entities, assumptions, or changes. In phylogenetics, for example, the preferred tree showing evolutionary relationships between species, molecules, or other entities is the one that requires the least amount of evolutionary change, that is, maximum parsimony.”

https://www.oxfordreference.com/view/10.1093/oi/authority.20110803100346221

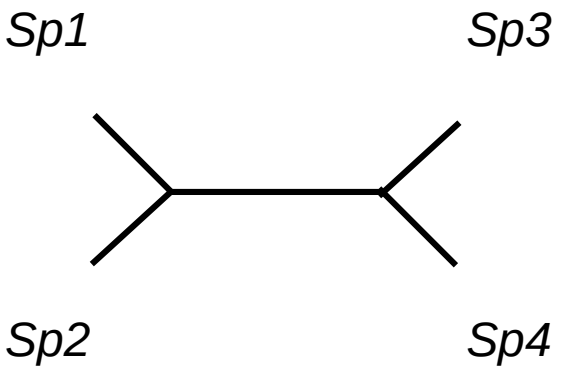
Maximum parsimony

- Has been advocated strongly by some against model-based approaches: many controversies (see “The Troubled Growth of Statistical Phylogenetics”, Felsenstein 2001)
- Edwards and Cavalli-Sforza (1963): the preferred evolutionary tree involves “the minimum net amount of evolution” = *Maximum parsimony tree*
- → For sequence data: find the phylogeny that involves the minimum number of substitutions

  → We need a way to count the minimum number of substitutions on a phylogeny = compute the **parsimony score** of a phylogenetic tree

Computing the parsimony score

```
Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...
```



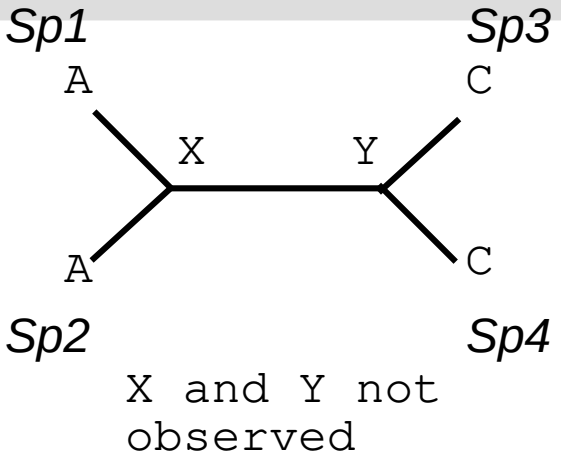
Parsimony score of a tree given an alignment: sum of the parsimony scores for each site

→ We assume that all sites are independent

Computing the parsimony score at one site

Species tree S

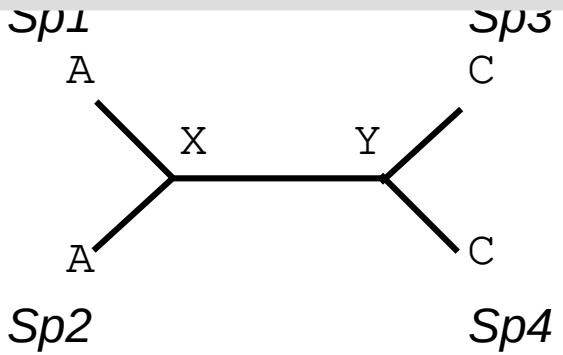
Site i



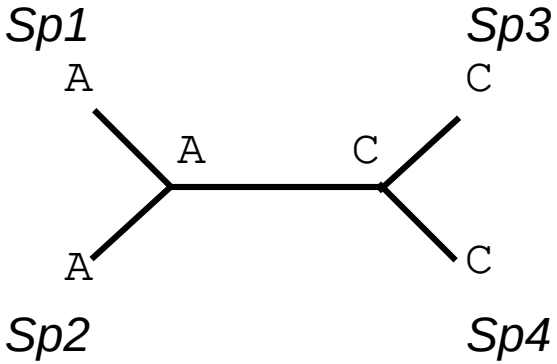
Computing the parsimony score at one site

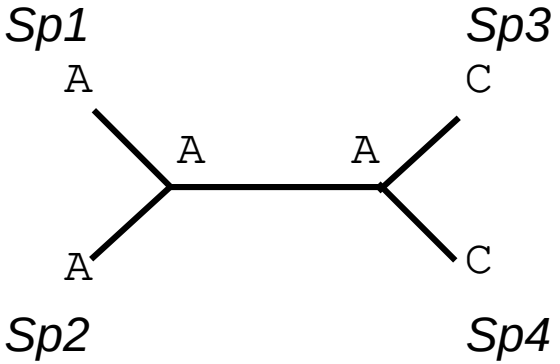
Species tree S

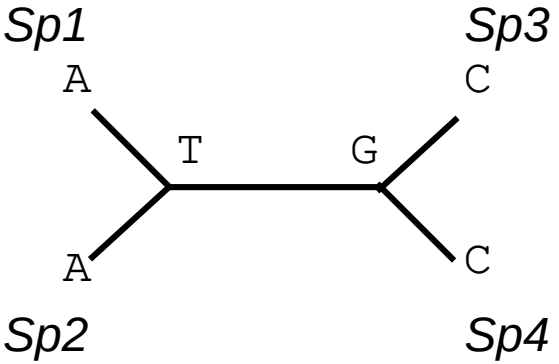
Site i



X and Y not observed



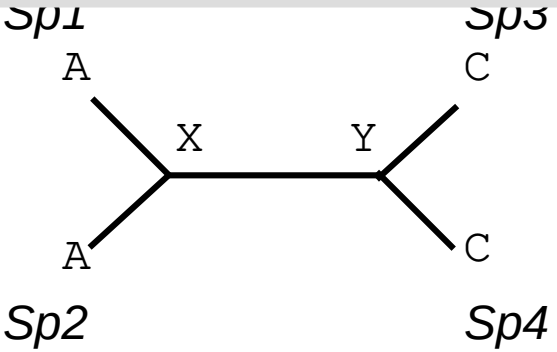




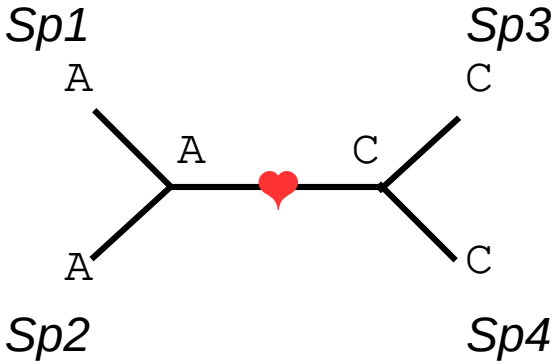
Computing the parsimony score at one site

Species tree S

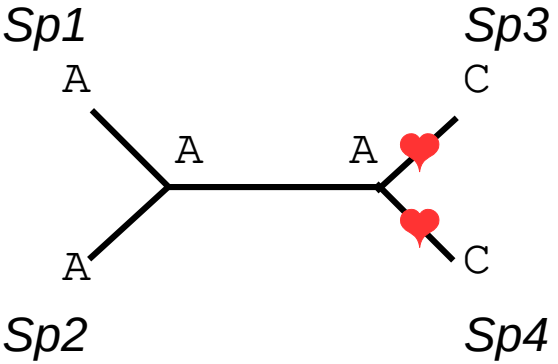
Site i



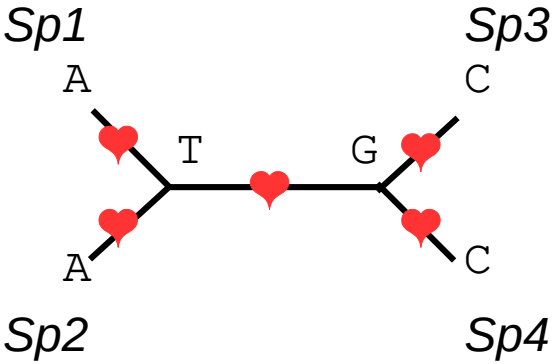
X and Y not observed



1 substitution



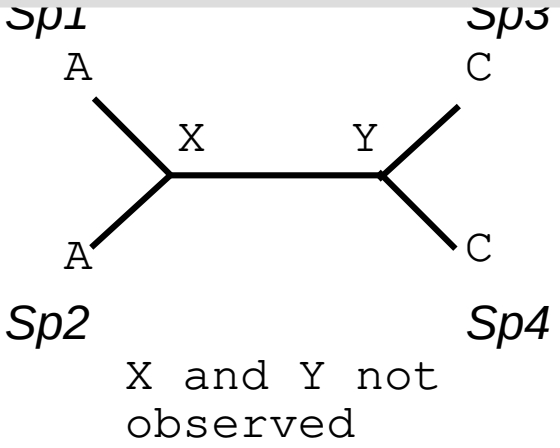
2 substitutions

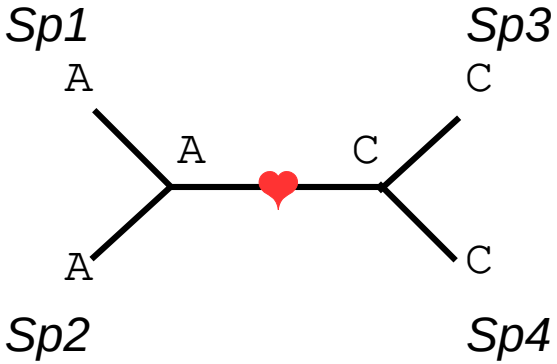


5 substitutions

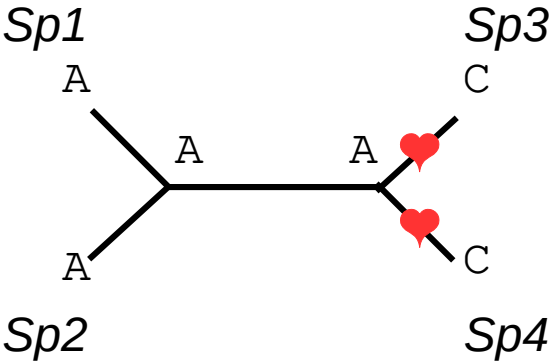
Computing the parsimony score at one site

Species tree S
Site i

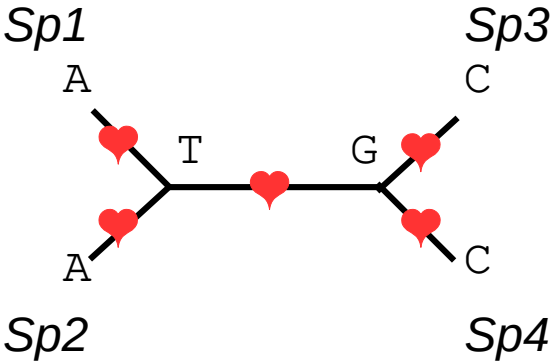




1 substitution



2 substitutions



5 substitutions

Parsimony score of species tree S for site i: 1

Computing the parsimony score of a tree at one site

- Naive brute force approach: test all possible assignments on internal nodes
  - N internal nodes, 4 possibilities {A,C,G,T} per node
  - → 4^N possibilities to try

- Fitch's algorithm (1971):
  - Arbitrarily root the tree
  - Compute, from the tips up, two elements per node:
    - P: The score of the underlying subtree
    - X: The set of states possible at that node, given the score P
  - Complexity: O(4N)

Computing the parsimony score of a tree at one site

- Naive brute force approach: test all possible assignments on internal nodes
  - N internal nodes, 4 possibilities {A,C,G,T} per node
  - → 4^N possibilities to try

For 2 internal nodes: $4^2 = 16$

- Fitch's algorithm (1971):
  - Arbitrarily root the tree
  - Compute, from the tips up, two elements per node:
    - P: The score of the underlying subtree
    - X: The set of states possible at that node, given the score P
  - Complexity: O(4N)

Computing the parsimony score of a tree at one site

- Naive brute force approach: test all possible assignments on internal nodes
  - N internal nodes, 4 possibilities {A,C,G,T} per node
  - → 4^N possibilities to try

For 2 internal nodes: $4^2 = 16$

- Fitch's algorithm (1971):
  - Arbitrarily root the tree
  - Compute, from the tips up, two elements per node:
    - P: The score of the underlying subtree
    - X: The set of states possible at that node, given the score P
  - Complexity: O(4N)

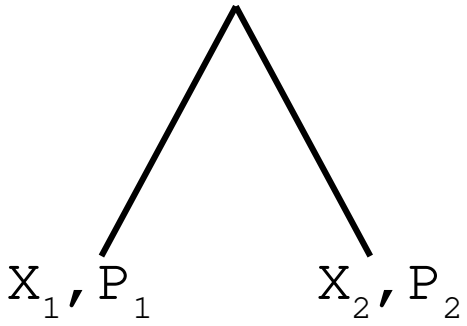
For 2 internal nodes: 4*2 =8

Fitch's algorithm

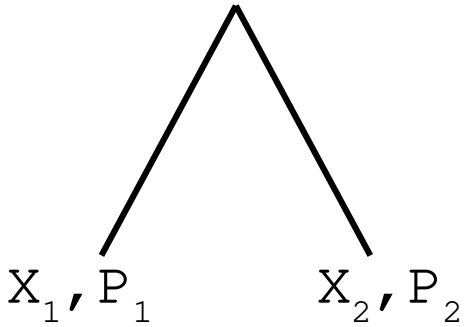
- P: The score of the underlying subtree
- X: The set of states possible at that node, given the score P

Climbing up the tree: computing P and X for a node given its children

1st case: $X_1 \cap X_2$ not empty



2nd case: $X_1 \cap X_2$ empty

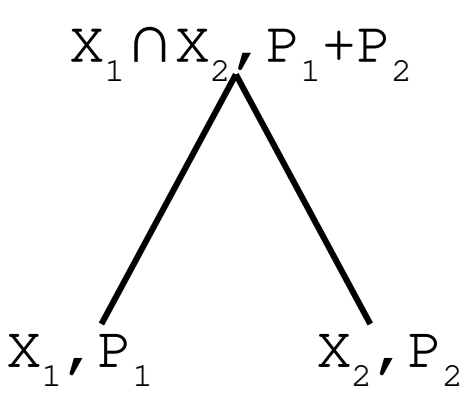


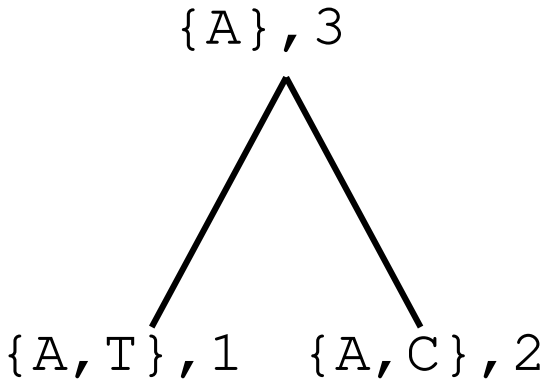
Fitch's algorithm

- P: The score of the underlying subtree
- X: The set of states possible at that node, given the score P

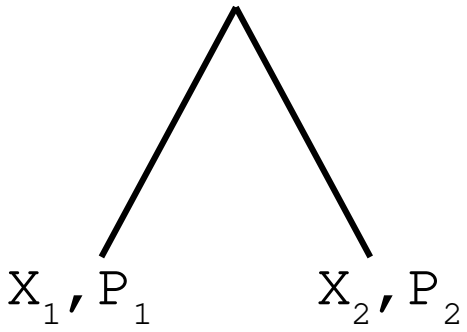
Climbing up the tree: computing P and X for a node given its children

1st case: $X_1 \cap X_2$ not empty





2nd case: $X_1 \cap X_2$ empty

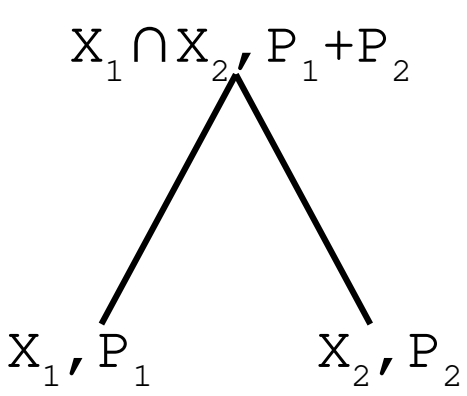


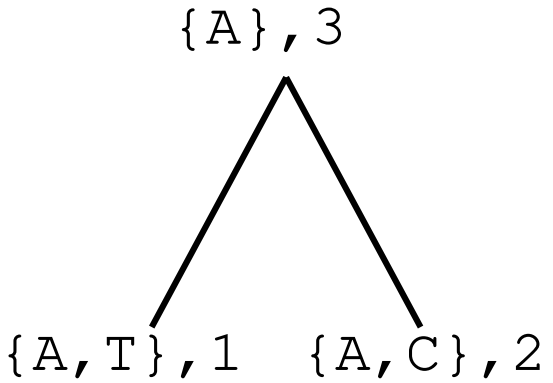
Fitch's algorithm

- P: The score of the underlying subtree
- X: The set of states possible at that node, given the score P

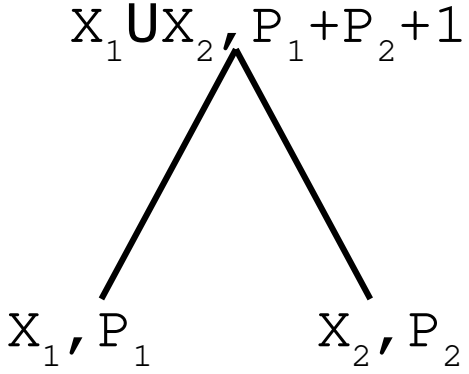
Climbing up the tree: computing P and X for a node given its children

1st case: $X_1 \cap X_2$ not empty





2nd case: $X_1 \cap X_2$ empty

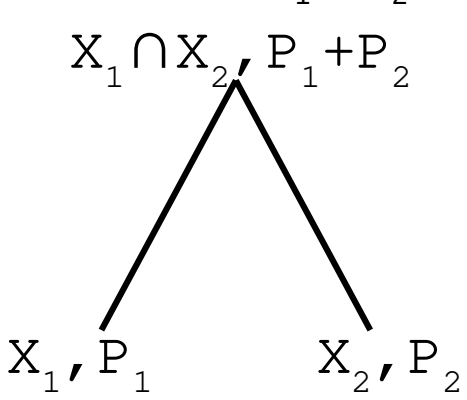


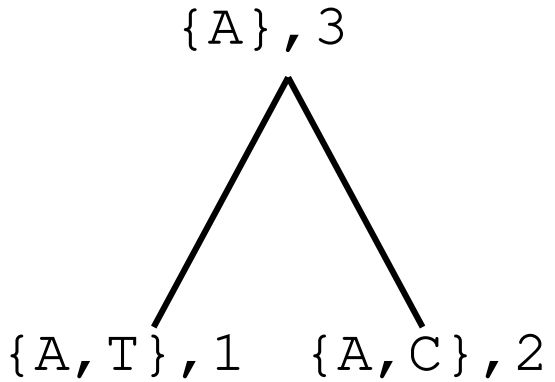
Fitch's algorithm

- P: The score of the underlying subtree
- X: The set of states possible at that node, given the score P

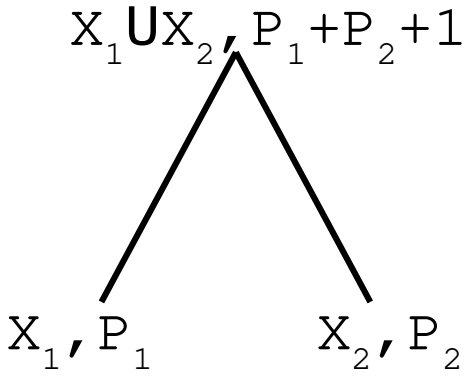
Climbing up the tree: computing P and X for a node given its children

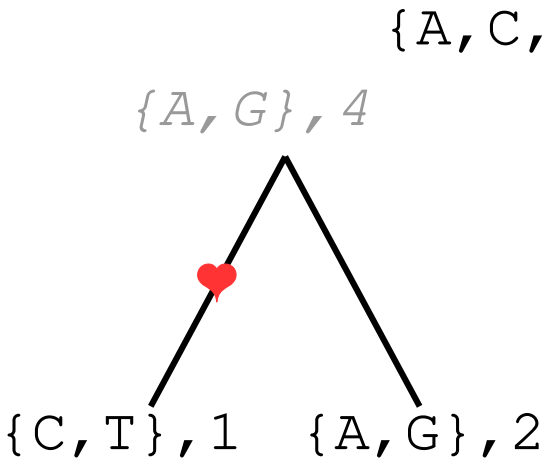
1st case: $X_1 \cap X_2$ not empty

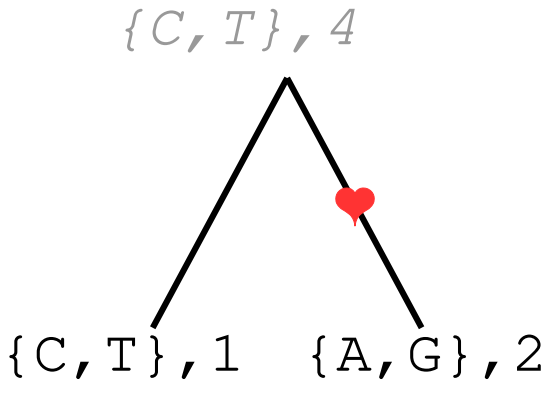




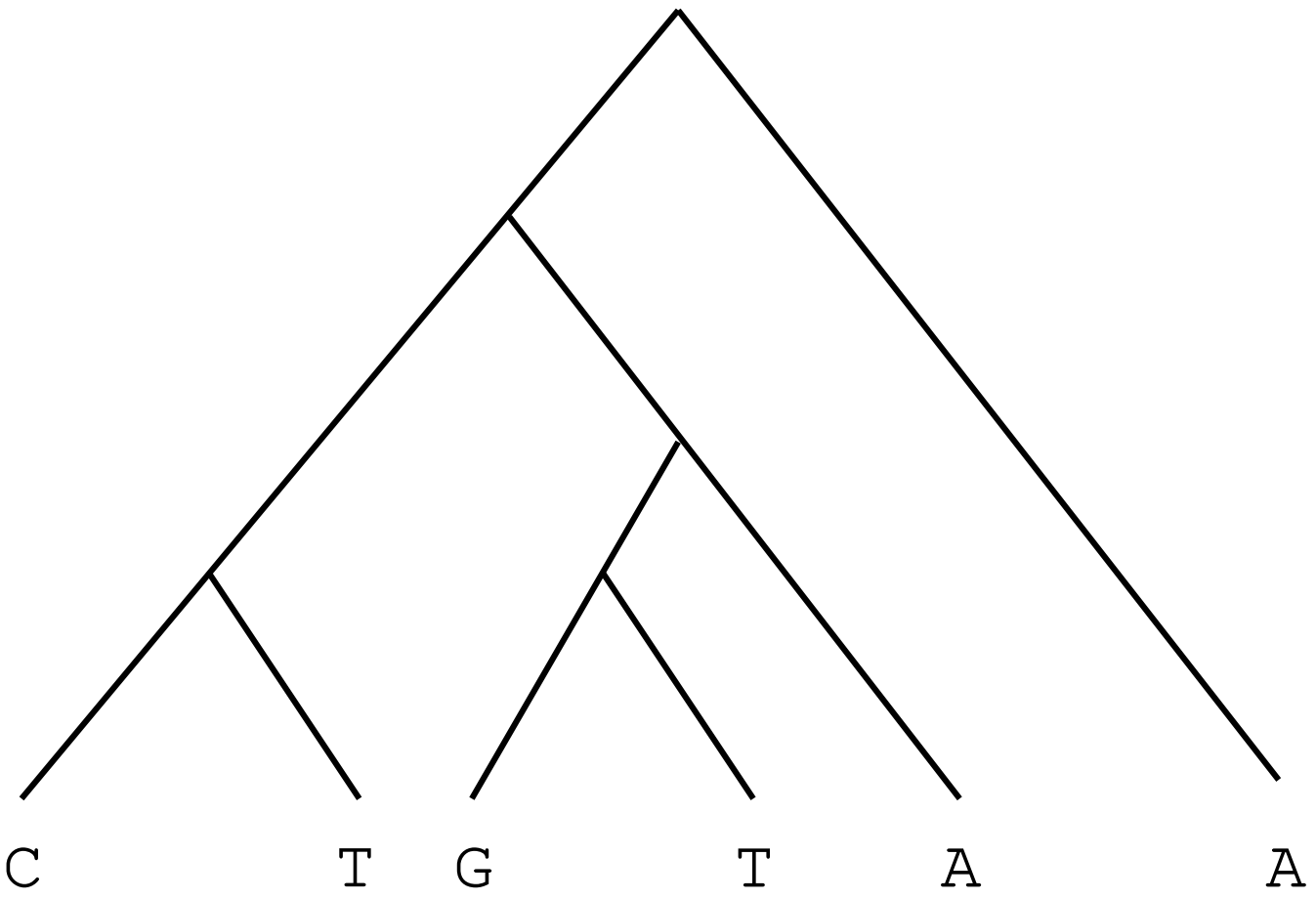
2nd case: $X_1 \cap X_2$ empty



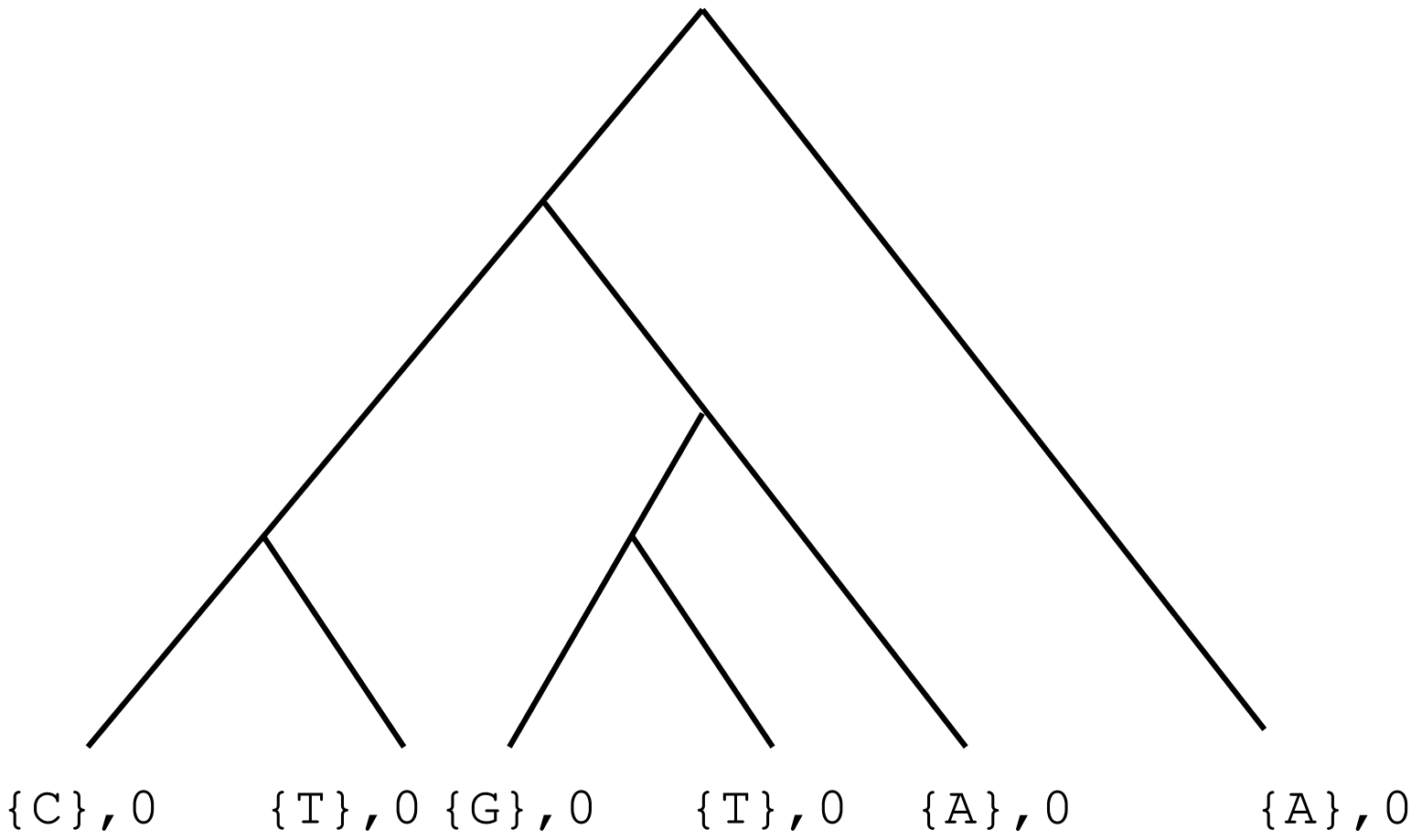




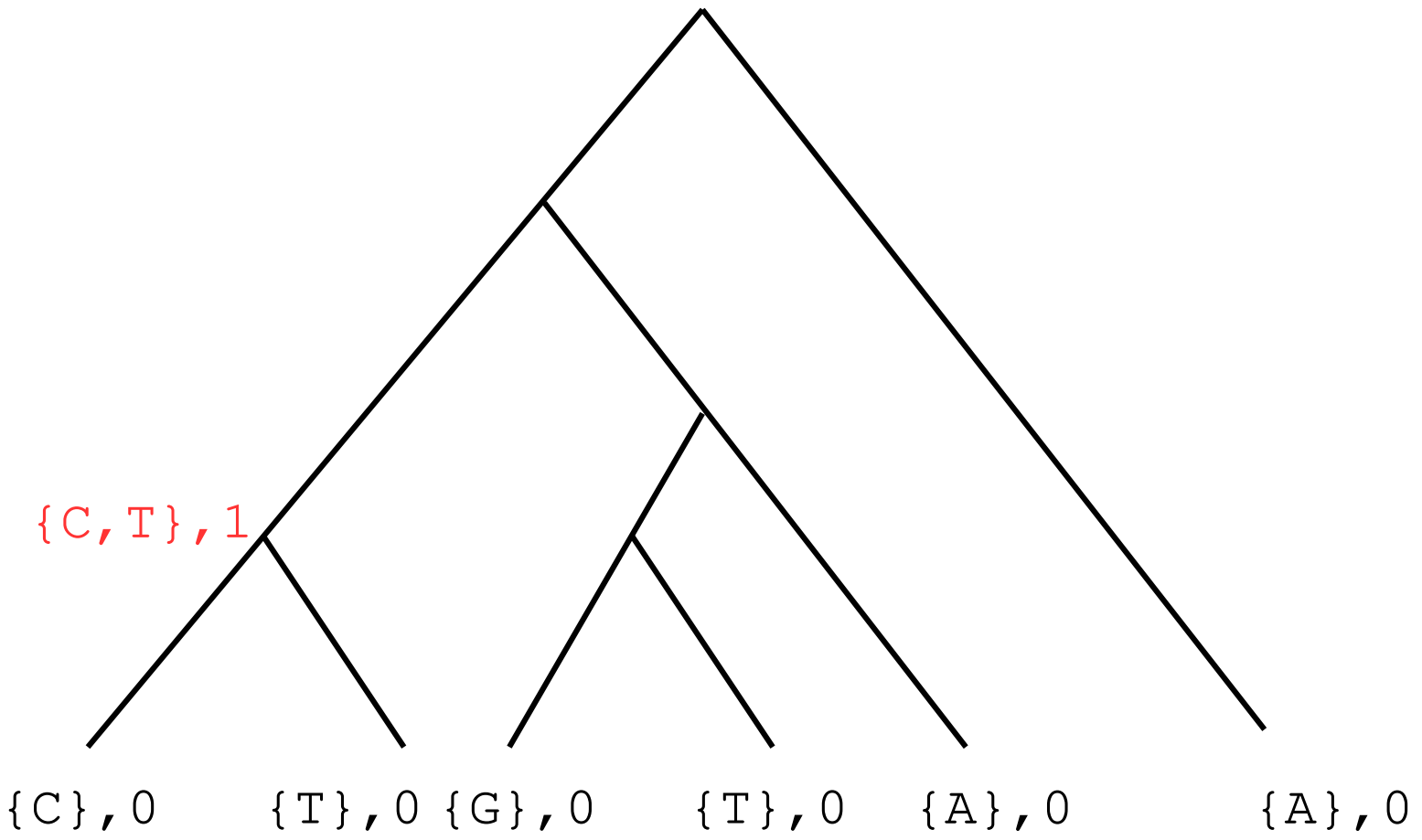
Example



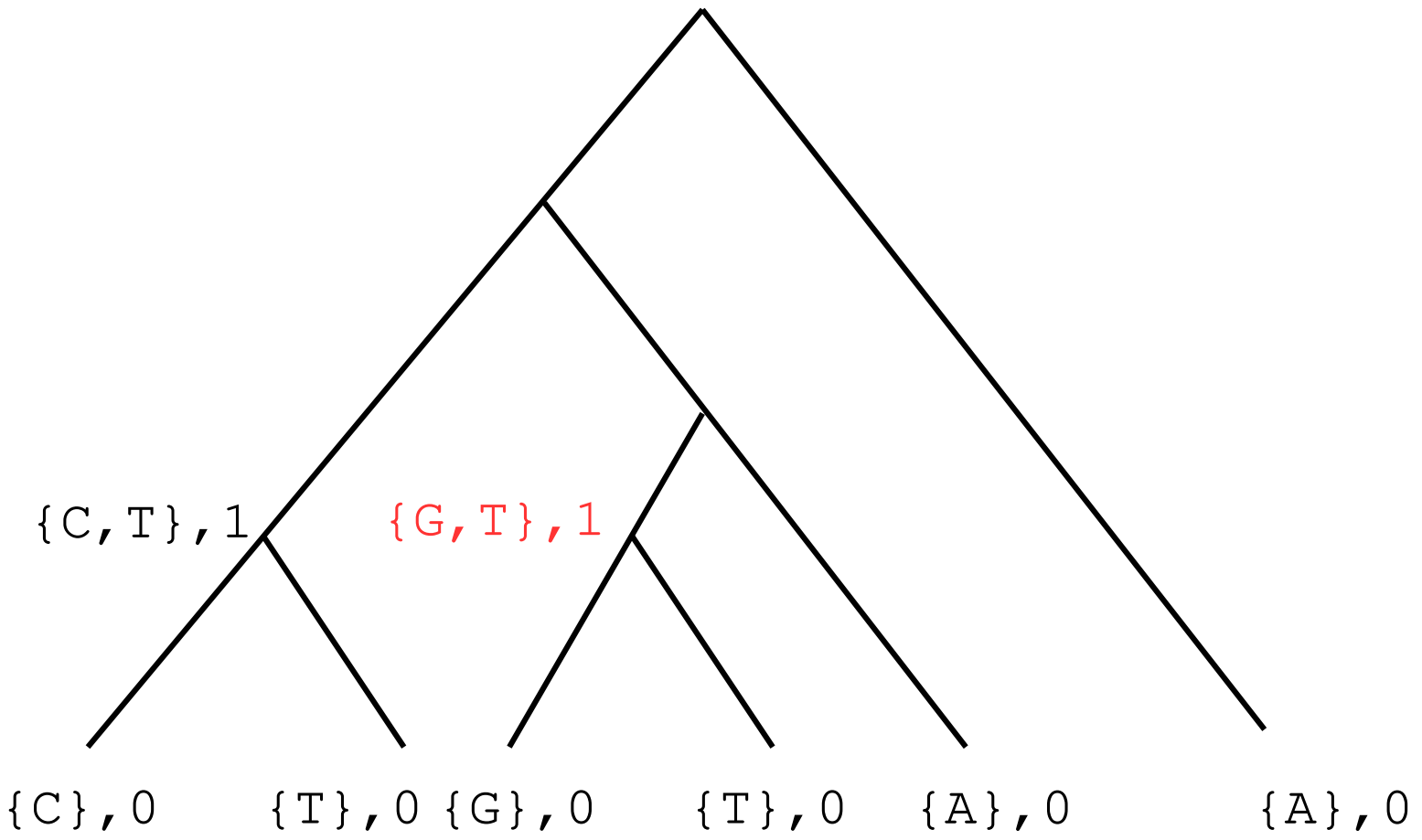
Example



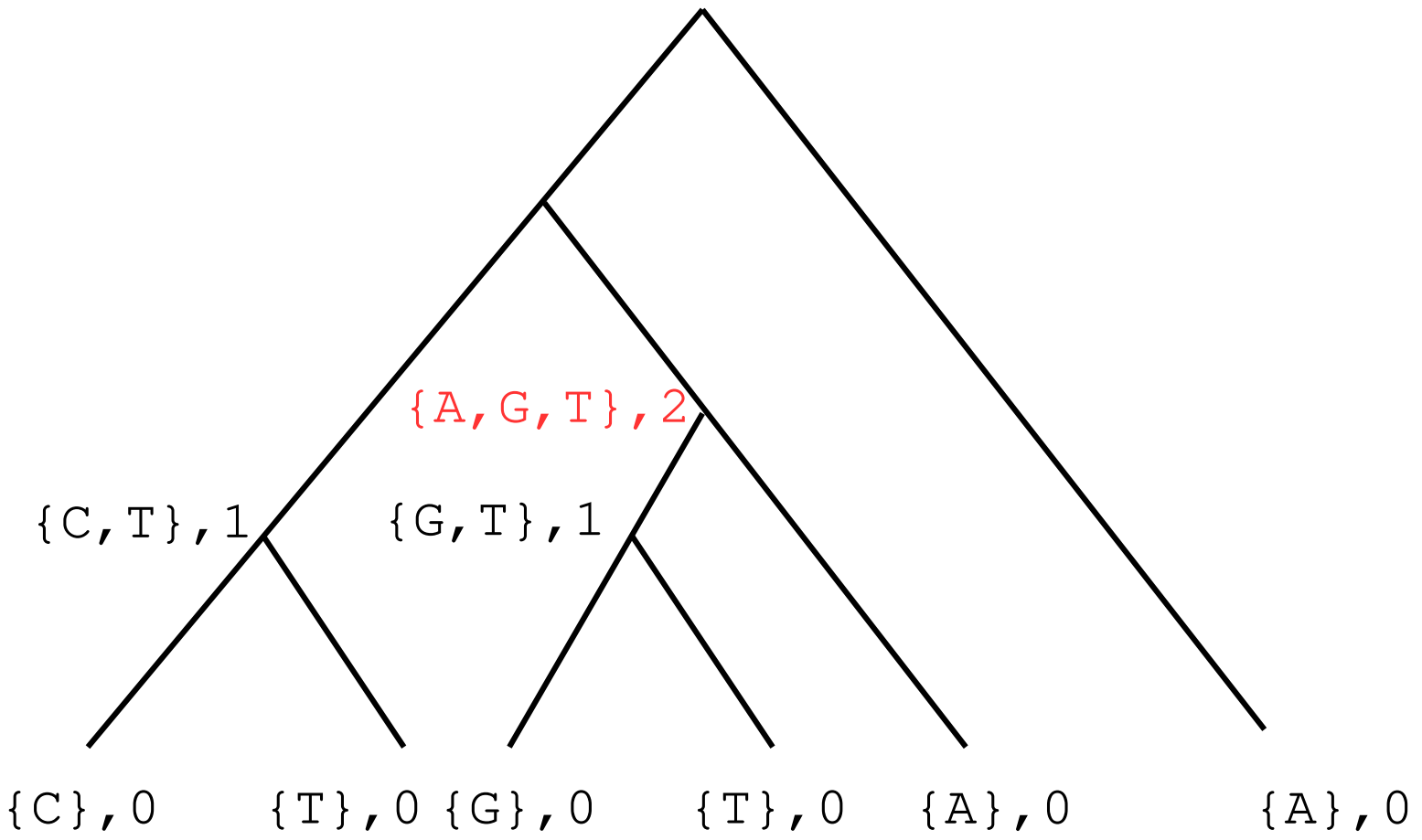
Example



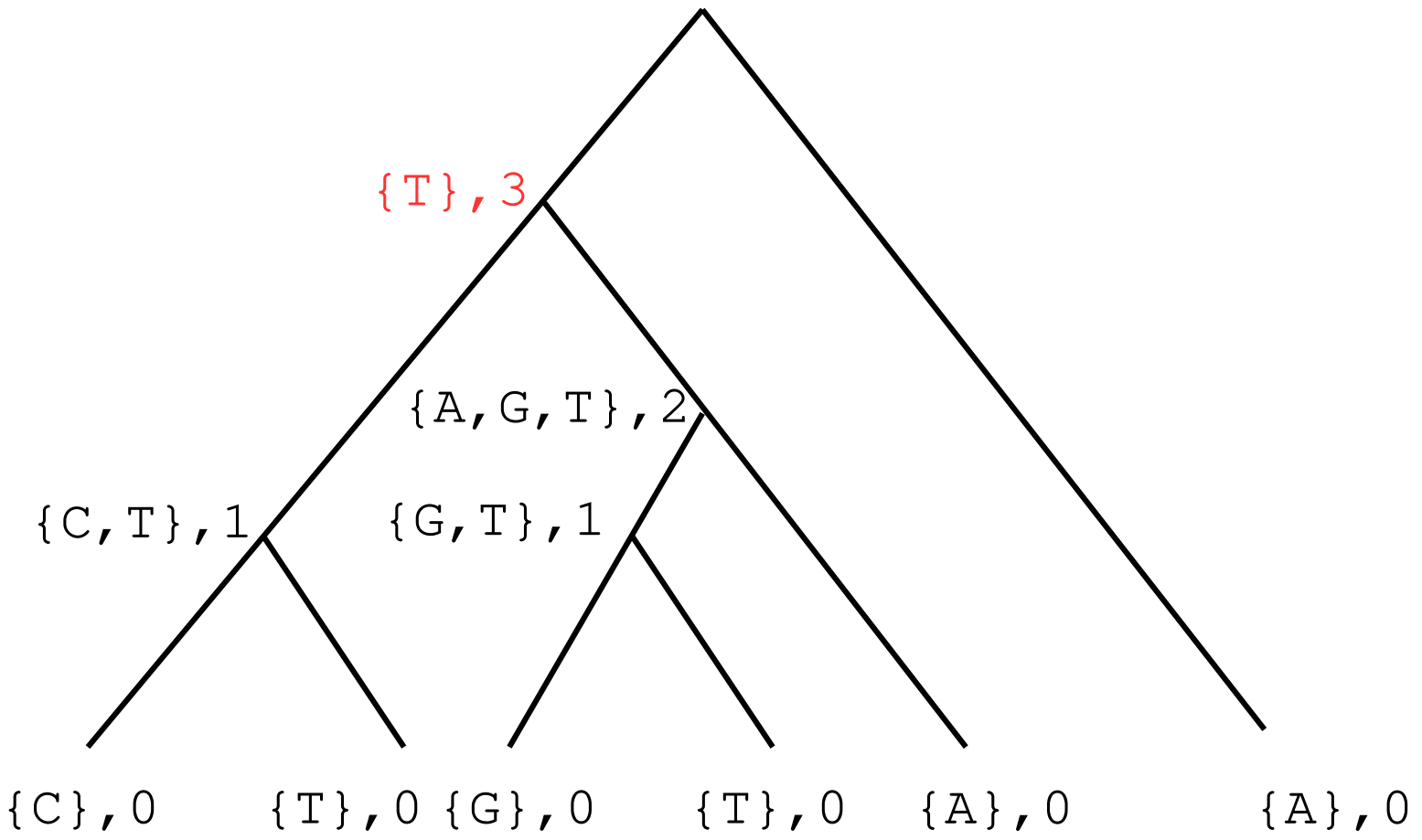
Example



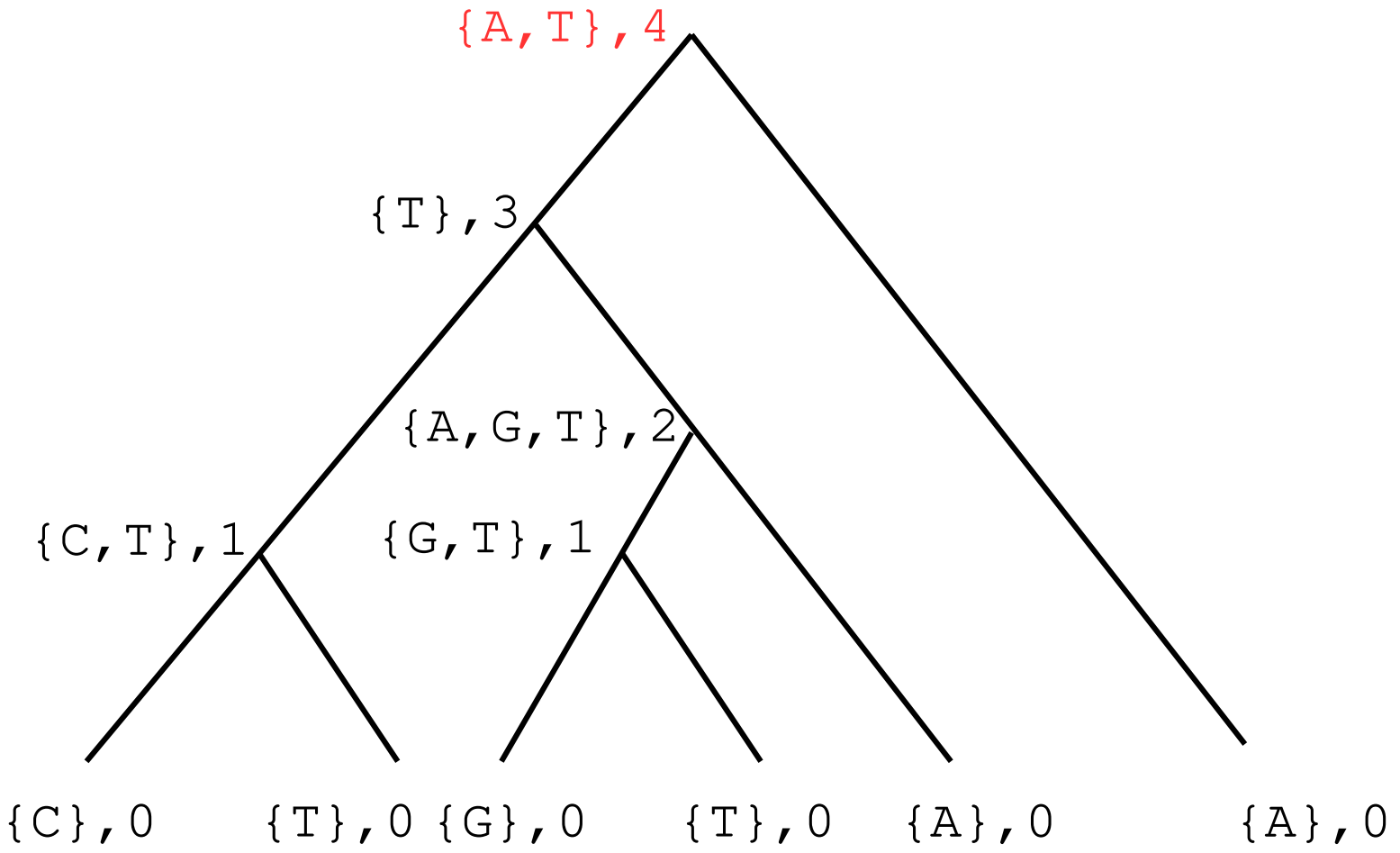
Example



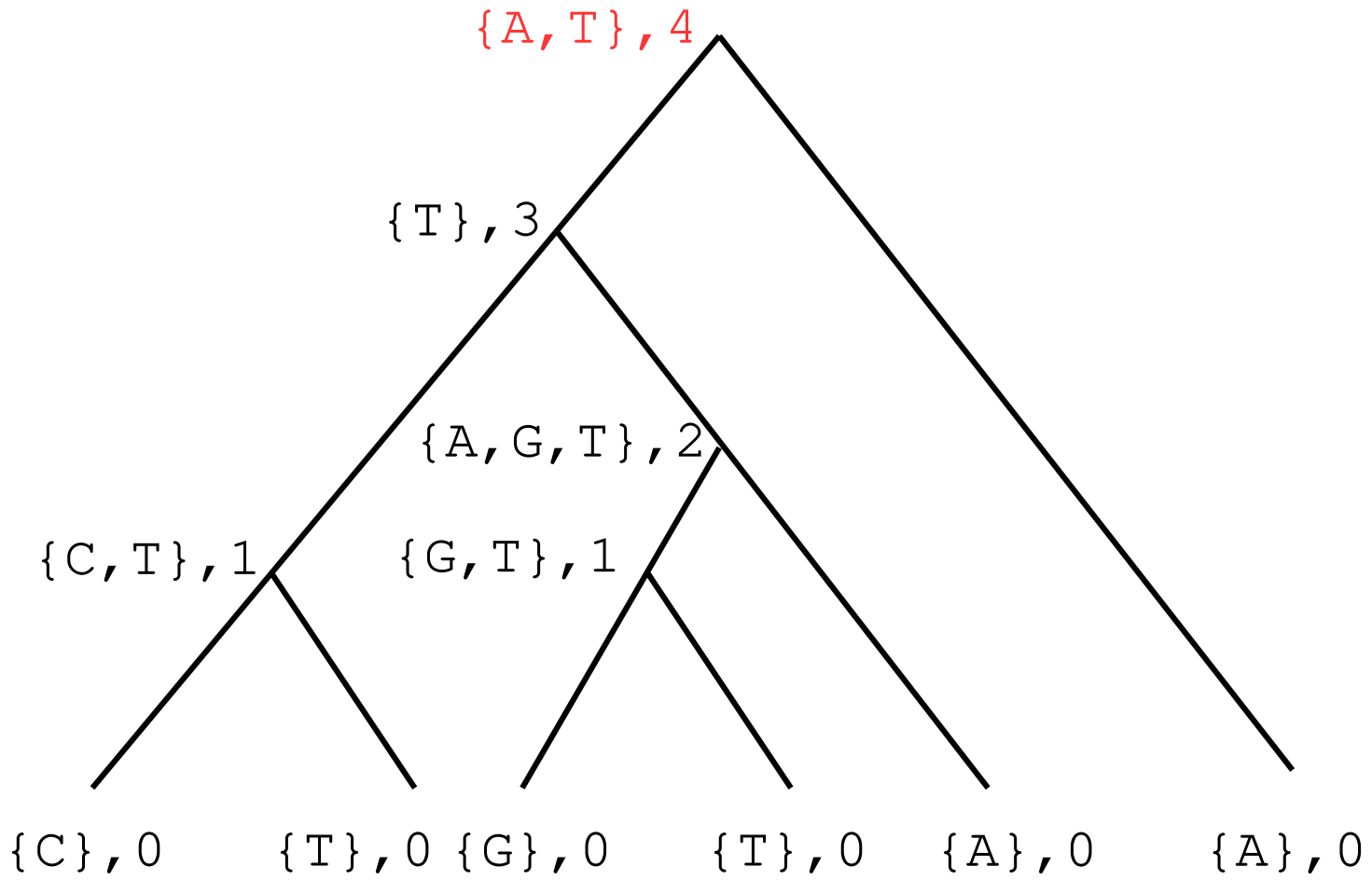
Example

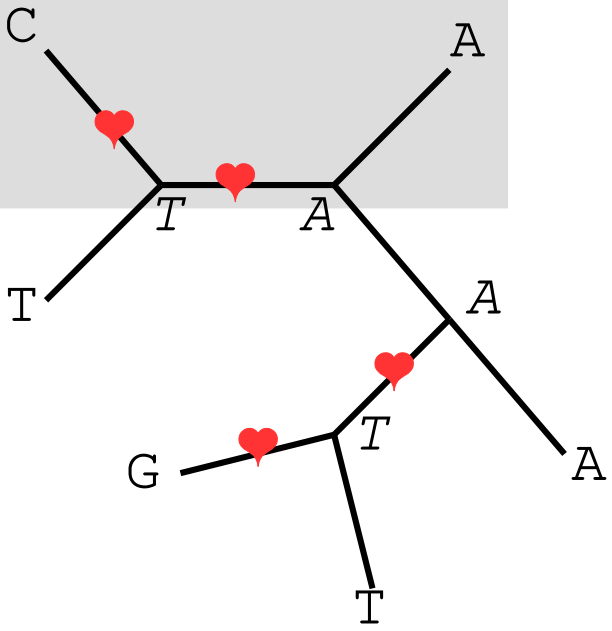


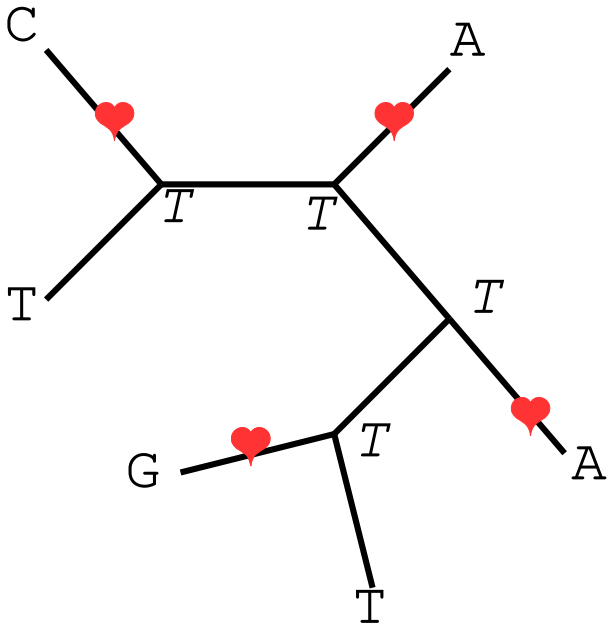
Example



Example







Several possible scenarios

Plan: Criteria for evaluating phylogenies

- Criteria for evaluating phylogenetic trees:
  - Parsimony
  - Distance methods
  - Maximum Likelihood
  - Posterior probability (Bayesian approach)
- Conventions:
  - We’re dealing with aligned sequence data
  - gaps are not taken into account

Distance methods

- Distance-based approaches:
  - least squares methods,
  - Minimum evolution method,
  - Neighbor Joining.

Minimum Evolution or least squares: distance methods

- Use a distance matrix:

Sp1 ATGCGCT...

Sp2 AGTCGCA...

Sp3 AGGTGCA...

Sp4 ATGCCCT...

Minimum Evolution or least squares: distance methods

- Uses a distance matrix:

Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

Minimum Evolution or least squares: distance methods

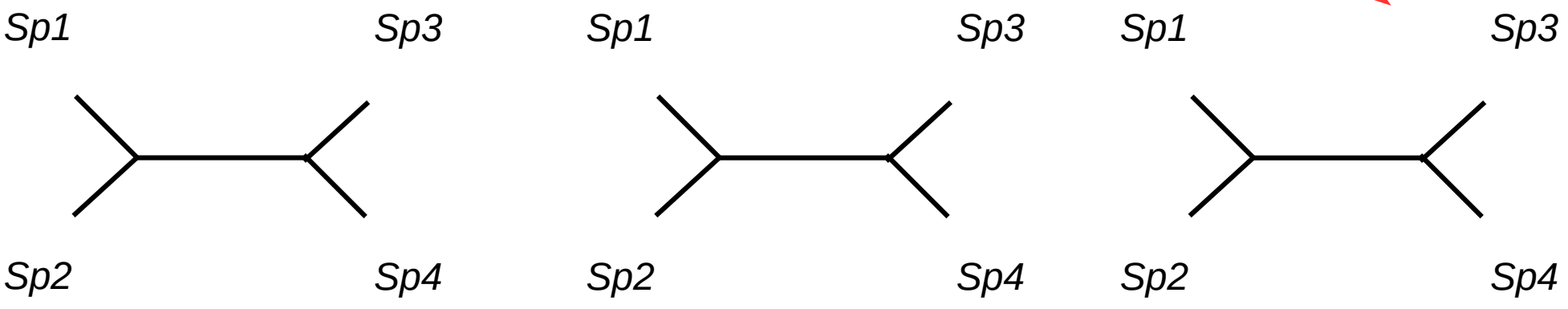
- Uses a distance matrix:

Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...

1

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

2



Minimum Evolution or least squares: distance methods

- Uses a distance matrix:

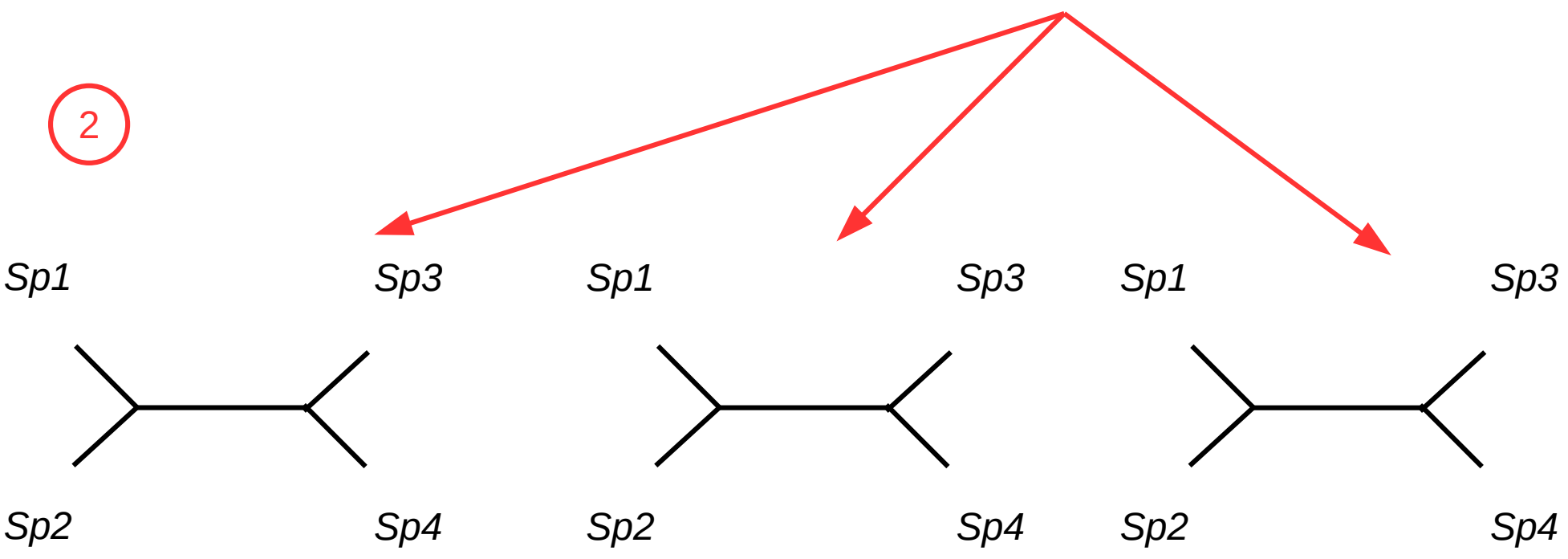
```
Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...
```

1

How to compute the distance matrix?

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

2



Minimum Evolution or least squares: distance methods

- Uses a distance matrix:

```
Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...
```

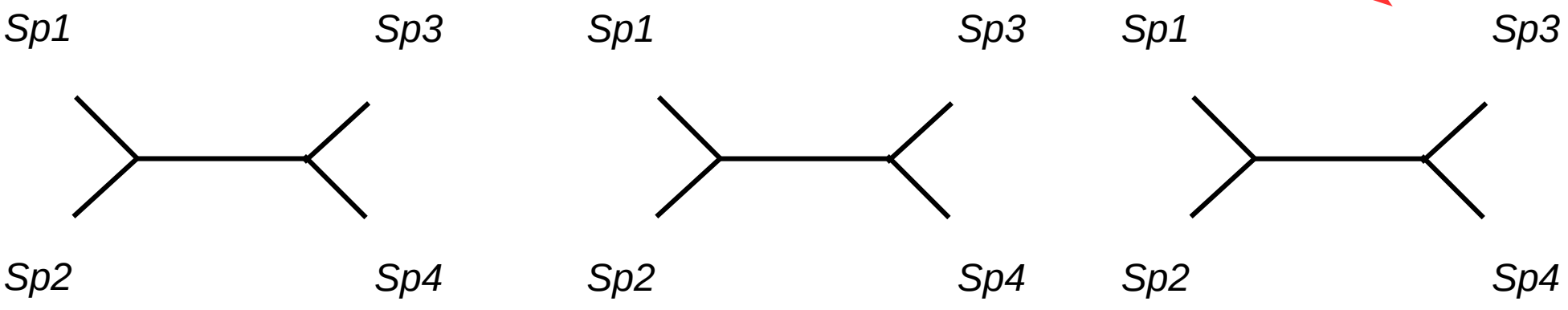
1

How to compute the distance matrix?

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

2

Which tree fits the distance matrix best?



1: How to compute distances between sequences?

- Simply count differences (observed divergence)

Sp1 ATGCGCT
Sp2 AGTCGCA ⟶ d(Sp1-Sp2) =

1: How to compute distances between sequences?

- Simply count differences (observed divergence)

```
Sp1 ATGCGCT
Sp2 AGTCGCA
      --   -
```

d(Sp1-Sp2) =

1: How to compute distances between sequences?

- Simply count differences (observed divergence)

```
Sp1 ATGCGCT
Sp2 AGTCGCA
      --   -
```

d(Sp1-Sp2) = 3/7~0.43

1: How to compute distances between sequences?

- Simply count differences (observed divergence)

```
Sp1 ATGCGCT
Sp2 AGTCGCA
      --   -
```

→ d(Sp1-Sp2) = 3/7~0.43

- Use a model of sequence evolution

  → *cf. talk on models*

  - Advantages:
    - *Hidden substitutions are taken into account*
    - *Parameters of the model of substitution can be estimated in the Maximum Likelihood framework*

Minimum Evolution or least squares: distance methods

- Uses a distance matrix:

```
Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...
```

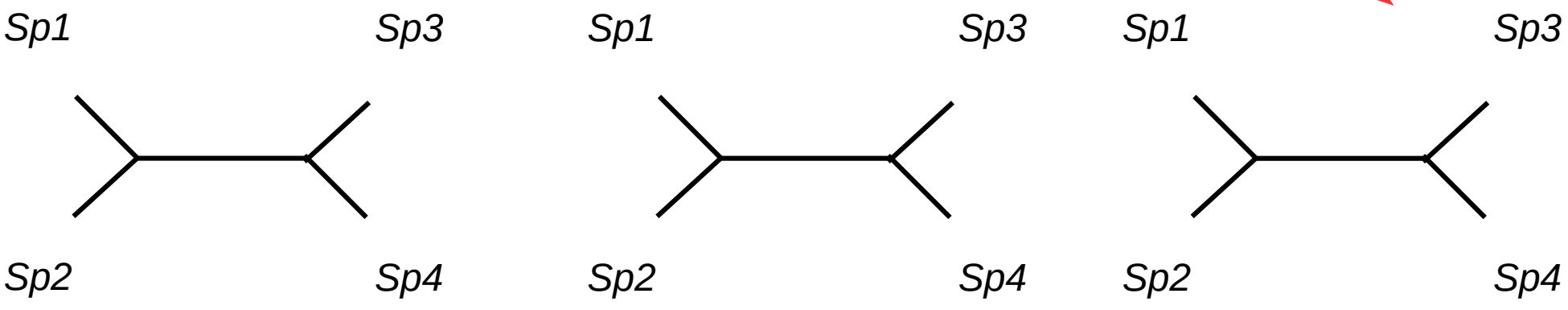
1

How to compute the distance matrix? ✓

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

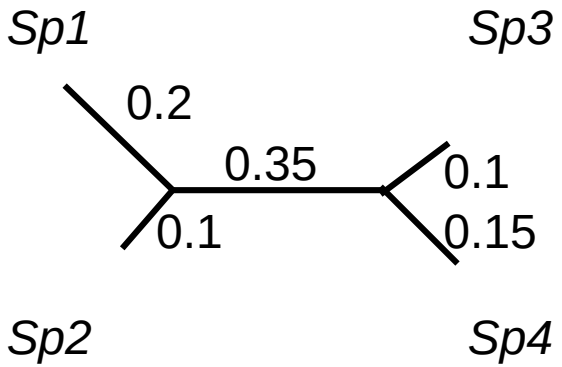
2

Which tree fits the distance matrix best?



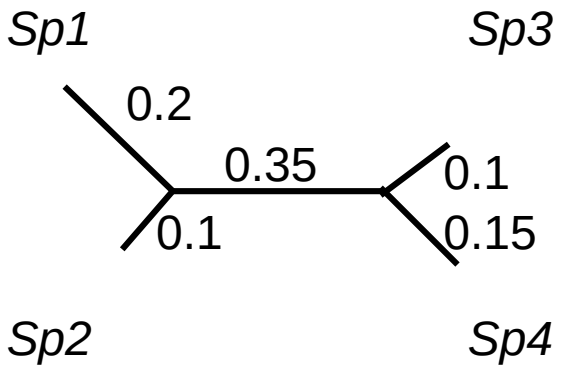
2: How to compute the fit between a distance matrix and a tree?

A tree implies distances between tips: compare those **patristic** distances to sequence-based distances



2: How to compute the fit between a distance matrix and a tree?

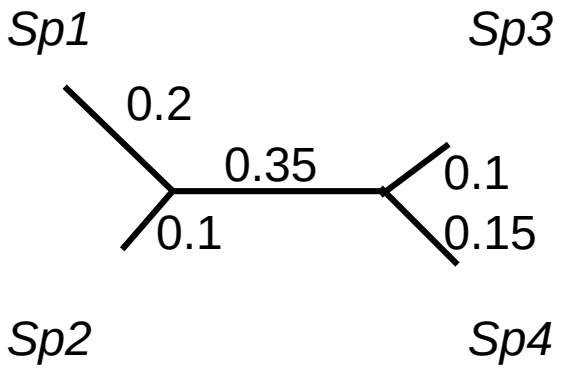
A tree implies distances between tips: compare those **patristic** distances to sequence-based distances



	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.3	0.65	0.7
Sp2	0.3	0	0.55	0.6
Sp3	0.65	0.55	0	0.25
Sp4	0.7	0.6	0.25	0

2: How to compute the fit between a distance matrix and a tree?

A tree implies distances between tips: compare those **patristic** distances to sequence-based distances

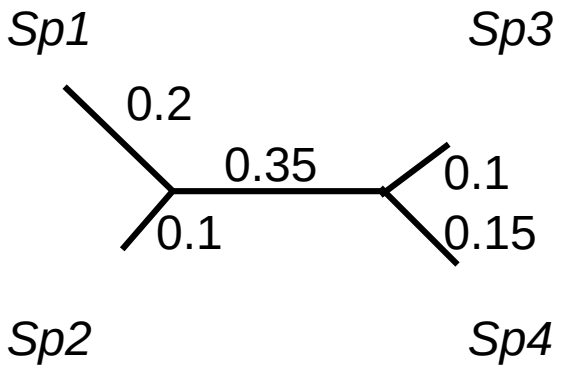


	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.3	0.65	0.7
Sp2	0.3	0	0.55	0.6
Sp3	0.65	0.55	0	0.25
Sp4	0.7	0.6	0.25	0

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

2: How to compute the fit between a distance matrix and a tree?

A tree implies distances between tips: compare those **patristic** distances to sequence-based distances



	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.3	0.65	0.7
Sp2	0.3	0	0.55	0.6
Sp3	0.65	0.55	0	0.25
Sp4	0.7	0.6	0.25	0

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

$score_{ULS} = (0-0)^2+(0.3-0.1)^2+(0.65-0.2)^2+\ldots$

With ULS: Unweighted Least Squares
(other criteria have been proposed)

Computing the optimal distances on a given topology

Using the ULS criterion, we can compute the fit between a sequence-based distance matrix and any tree (topology + branch lengths), thanks to the patristic matrix trick.

Computing the optimal distances on a given topology

Using the ULS criterion, we can compute the fit between a sequence-based distance matrix and any tree (topology + branch lengths), thanks to the patristic matrix trick.

But how can we pick branch lengths on the topology?

Computing the optimal distances on a given topology

Using the ULS criterion, we can compute the fit between a sequence-based distance matrix and any tree (topology + branch lengths), thanks to the patristic matrix trick.

But how can we pick branch lengths on the topology?

ULS provides a mathematical way to find the optimal branch lengths on a given topology! This involves some simple matrix algebra (solving a set of linear equations).

Searching for the best tree using Unweighted Least Squares

- We now know how to compute the ULS score of a tree topology. It involves:
  - Matrix algebra to find the best branch lengths
  - Computing the $score_{ULS}$ for that tree
- Given a set of tree topologies, we can compute the “best” tree topology according to the ULS criterion: it is the one with the lowest $score_{ULS}$
- How to obtain a set of tree topologies to score is tackled later in the course *(see Alexis’s talk)*

Searching for the best tree using Unweighted Least Squares

- We now know how to compute the ULS score of a tree topology. It involves:
  - Matrix algebra to find the best branch lengths
  - Computing the $score_{ULS}$ for that tree
- Given a set of tree topologies, we can compute the “best” tree topology according to the ULS criterion: it is the one with the lowest $score_{ULS}$
- How to obtain a set of tree topologies to score is tackled later in the course *(see Alexis’s talk)*

Minimum evolution criterion

- Motivation similar to parsimony
- **Hypothesis:** the true tree should be the shortest tree
- → Idea:
  - Given a matrix of pairwise distances and a set of tree topologies to evaluate
  - Match pairwise distances onto each tree topology
  - Sum the branch lengths on each tree
  - ***Your best estimate is the tree with the smallest sum of branch lengths***

Minimum evolution criterion

- Motivation similar to parsimony
- **Hypothesis**: the true tree should be the shortest tree
- → Idea:
  - Given a matrix of pairwise distances and a set of tree topologies to evaluate
  - Match pairwise distances onto each tree topology: *Use least-squares fitting!*
  - Sum the branch lengths on each tree
  - ***Your best estimate is the tree with the smallest sum of branch lengths***

Minimum Evolution or least squares: distance methods

- Uses a distance matrix:

```
Sp1 ATGCGCT...
Sp2 AGTCGCA...
Sp3 AGGTGCA...
Sp4 ATGCCCT...
```

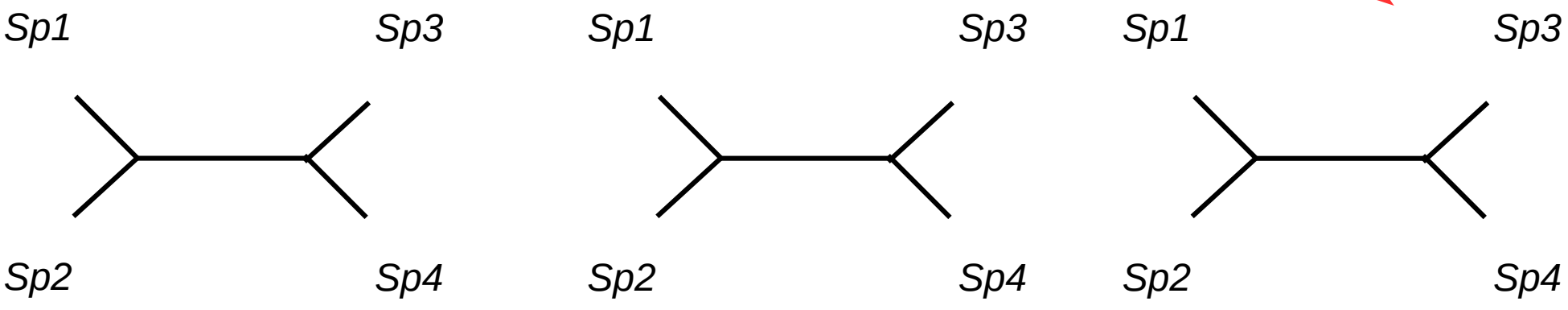
1

How to compute the distance matrix?

	Sp1	Sp2	Sp3	Sp4
Sp1	0	0.1	0.2	0.15
Sp2	0.1	0	0.3	0.01
Sp3	0.2	0.3	0	0.6
Sp4	0.15	0.01	0.6	0

2

ME: involves the patristic matrix, matrix algebra and summing branch lengths



Minimum evolution criterion

- To obtain a Minimum Evolution tree, at some point we have to use Least Squares estimation to assign branch lengths to a tree topology

  → hybrid approach where two different criteria are mixed up
- However, Minimum evolution works pretty well in practice
- Neighbor-Joining (Saitou and Nei, 1987) is a famous heuristic algorithm for finding the Minimum Evolution tree (not seen in our course, but has been very widely used); see Gascuel and Steel, 2006 for a clear explanation

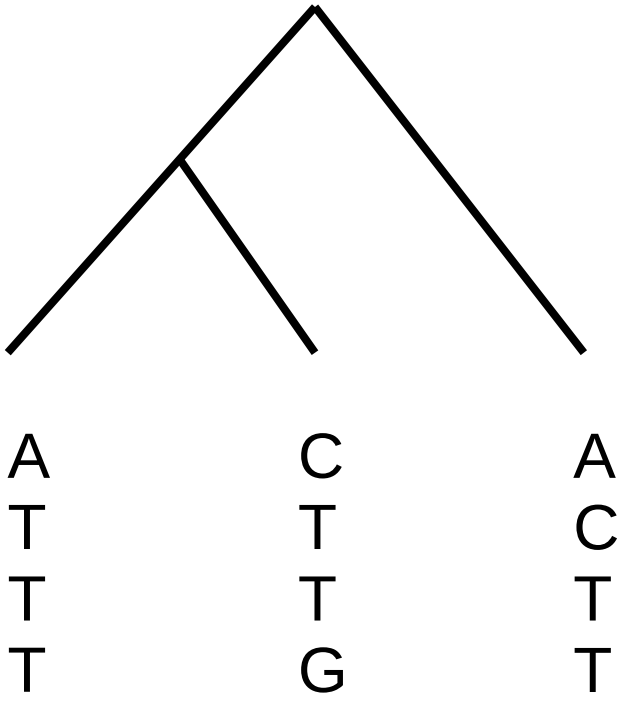
Summary on distance methods

- Distance methods are the fastest phylogenetic methods available, notably thanks to Neighbor Joining and others (e.g. BioNJ, Weighbor, FastME…)
- Can be based on models of sequence evolution to compute pairwise distances
- Better than Maximum Parsimony when sequences are divergent, but less accurate than Maximum Likelihood or Bayesian Inference
- The main reason is that distance methods do not use the entire data matrix together, but look at it pair of sequences by pair of sequences

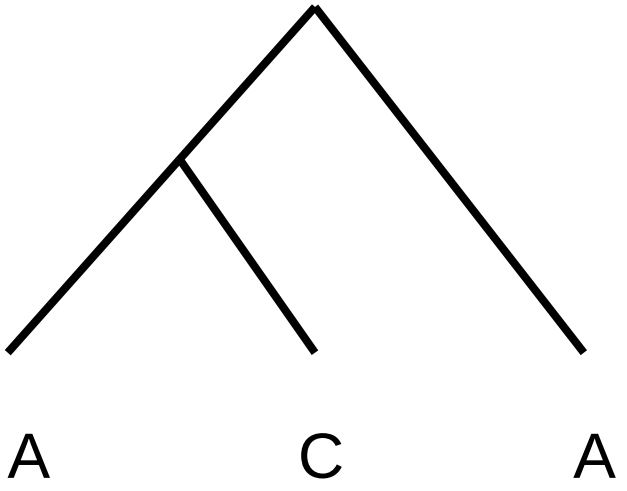
Plan: Criteria for evaluating phylogenies

- Criteria for evaluating phylogenetic trees:
  - Parsimony
  - Distance methods
  - Maximum Likelihood
  - Posterior probability (Bayesian approach)
- Conventions:
  - We’re dealing with aligned sequence data
  - gaps are not taken into account

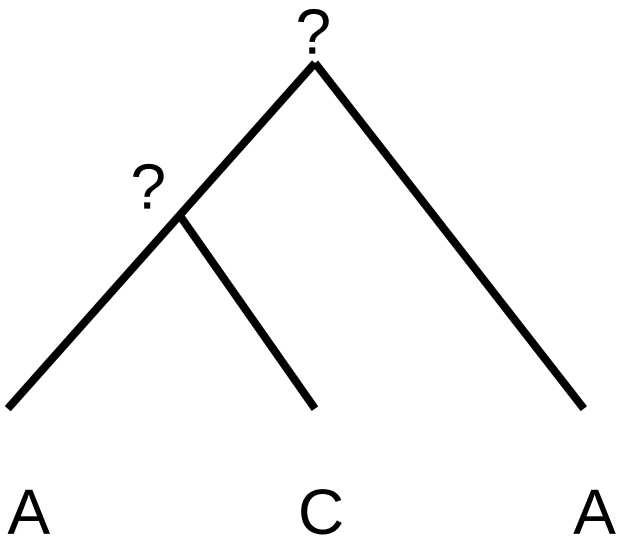
How to compute the likelihood of an alignment?



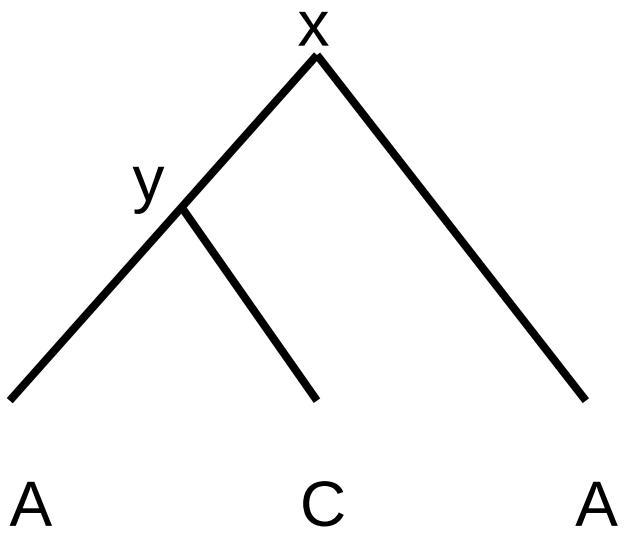
How to compute the likelihood of an alignment?



How to compute the likelihood of an alignment?

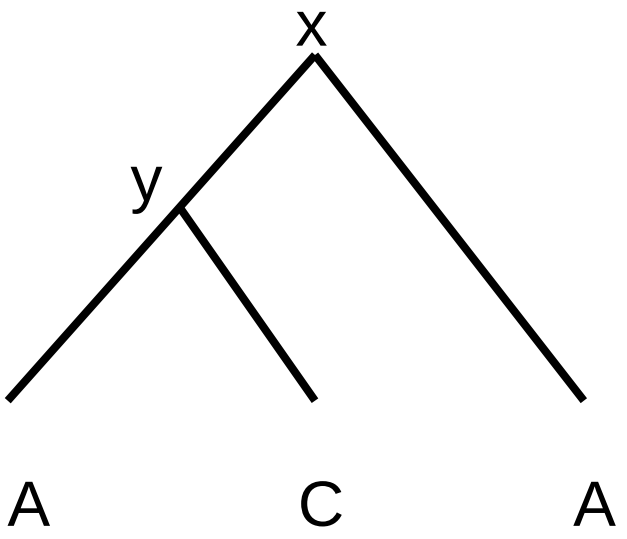


How to compute the likelihood of an alignment?



$x \in \{A,C,G,T\}$
$y \in \{A,C,G,T\}$

How to compute the likelihood of an alignment?



How can we compute
P({A,C,A} | model) ?

End-conditioned simulation

- Simulating along a branch that ends in state G:

How can we compute P(ending in G | model) ?

End-conditioned simulation

- Simulating along a branch that ends in state G:

  - Draw an initial state from a Multinomial distribution:

```
p=c(0.25, 0.25, 0.25, 0.25); state=rmultinom(n=1, p=p, size=1)
```

  - $t = t_0$; $N = 0$; $\lambda=0.1$
  - While $t < T$:
    - Draw from an exponential distribution a waiting time X_i until the next event; $t = t + X_i$
      - If $t < T$, change the state of the variable: `state=rmultinom(n=1, p=p, size=1)`
      - (Else ($t \geq T$): we stop)
  - If EndState != G: Failure
  - If EndState == G: Success

Can we use this to compute P(ending in G | model) ?

End-conditioned simulation

P(ending in G):

- **Nsuccess=0**
- For i in 1:10000:
  - Draw an initial state from a Multinomial distribution:

```
p=c(0.25, 0.25, 0.25, 0.25); state=rmultinom(n=1, p=p, size=1)
```

  - $t = t_0$; $N = 0$; $\lambda=0.1$
  - While $t < T$:
    - Draw from an exponential distribution a waiting time X_i until the next event; $t = t + X_i$
    - If $t < T$, change the state of the variable: `state=rmultinom(n=1, p=p, size=1)`
    - (Else ($t \geq T$): we stop)
  - **If EndState == G: Nsuccess +=1**

End-conditioned simulation

P(ending in G):

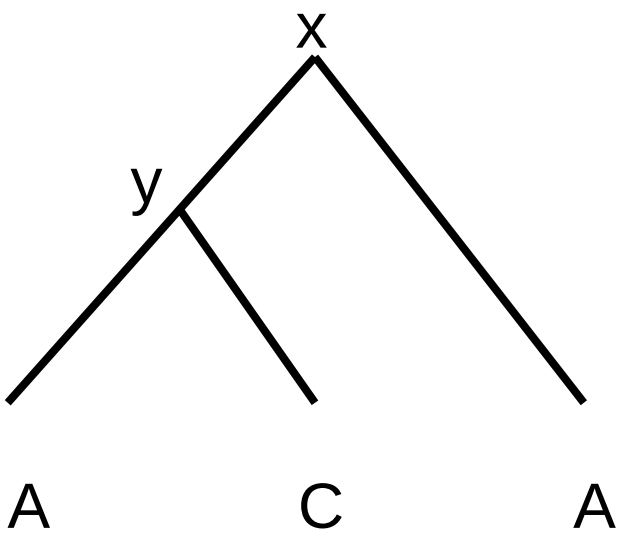
- **Nsuccess=0**
- For i in 1:10000:
  - Draw an initial state from a Multinomial distribution:

```
p=c(0.25, 0.25, 0.25, 0.25); state=rmultinom(n=1, p=p, size=1)
```

  - $t = t_0$; $N = 0$; $\lambda=0.1$
  - While $t < T$:
    - Draw from an exponential distribution a waiting time X_i until the next event; $t = t + X_i$
    - If $t < T$, change the state of the variable: `state=rmultinom(n=1, p=p, size=1)`
    - (Else ($t \geq T$): we stop)
  - **If EndState == G: Nsuccess +=1**

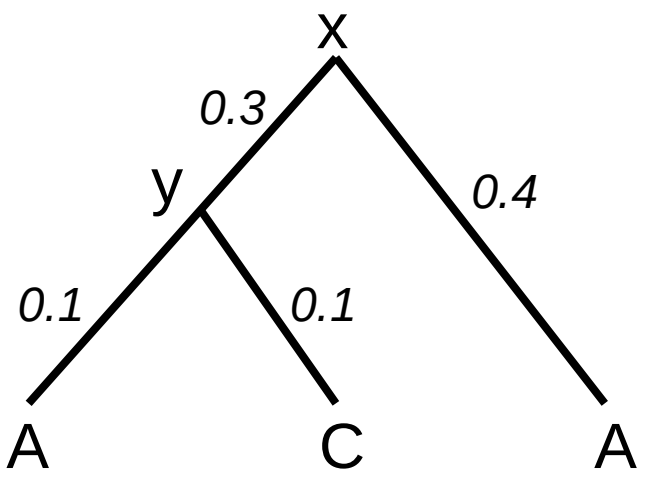
→ ***P(ending in G) = Nsuccess/10000***

End-conditioned simulation on a tree



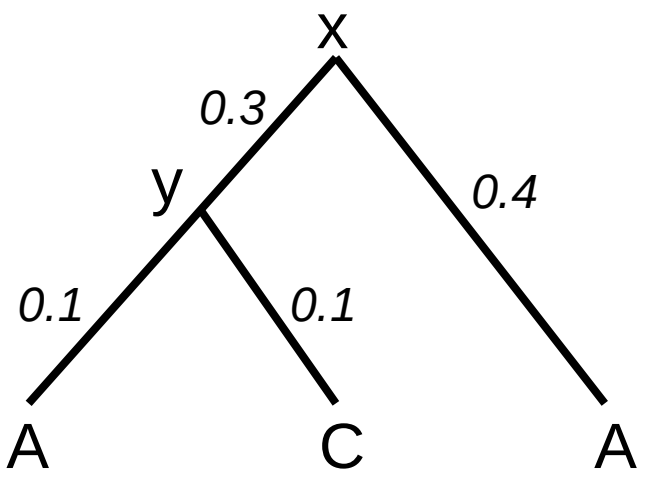
How can we compute
P({A,C,A} | model) ?

Computing P({A,C,A}|model) by simulation



- Nsuccess = 0
- Repeat 10 000 times:
  - Randomly pick x ϵ {A,C,G,T}
  - Simulate along branch xy (length 0.3)
  - Simulate along branch yA (length 0.1)
  - Simulate along branch yC (length 0.1)
  - Simulate along branch xA (length 0.4)
  - If we have ACA at the tips: Nsuccess++
- P({A,C,A}|model) = Nsuccess/10000

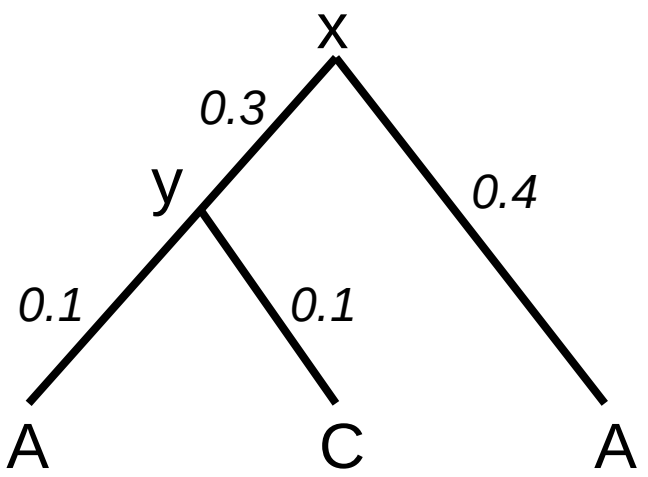
Computing P({A,C,A}|model) by simulation



- Nsuccess = 0
- Repeat 10 000 times:
  - Randomly pick x ϵ {A,C,G,T}
  - Simulate along branch xy (length 0.3)
  - Simulate along branch yA (length 0.1)
  - Simulate along branch yC (length 0.1)
  - Simulate along branch xA (length 0.4)
  - If we have ACA at the tips: Nsuccess++
- P({A,C,A}|model) = Nsuccess/10000

Try and implement this in R!

Computing P({A,C,A}|model) by simulation



- Nsuccess = 0
- Repeat 10 000 times:
  - Randomly pick x ϵ {A,C,G,T}
  - Simulate along branch xy (length 0.3)
  - Simul
  - Simulate along branch yC (length 0.1)
  - Simulate along branch xA (length 0.4)
  - If we have ACA at the tips: Nsuccess++
- P({A,C,A}|model) = Nsuccess/10000

http://rpubs.com/boussau/simuDNALikelihood

Interpretation

- We generate a sample of substitution histories along the tree, given the model
- Among those histories, we count those that are compatible with the data at the tips
- This proportion is the probability of the data given the model, the *likelihood:*

$$P(\text{site pattern}|\text{model})$$

An inefficient approach

- This approach works well for 1 site and 3 tips
- How can we scale to more tips?

→ *Use an analytical approach to integrate efficiently over all substitution histories*

2 elements:

- *Integrate over all substitution histories along a branch*
- *Integrate over all states at internal nodes*

An inefficient approach

- This approach works well for 1 site and 3 tips
- How can we scale to more tips?

→ *Use an analytical approach to integrate efficiently over all substitution histories*

2 elements:

- *Integrate over all substitution histories along a branch*
- *Integrate over all states at internal nodes*

Cf Alexis's talk!

Integrating over substitution histories along a branch

- Given a vector of initial frequencies F(0) = {A(0), C(0), G(0), T(0)}, how can we compute the vector of frequencies at time t F(t)?

Integrating over substitution histories along a branch

- Given a vector of initial frequencies F(0) = {A(0), C(0), G(0), T(0)}, how can we compute the vector of frequencies at time t F(t)?

  For an infinitesimal dt:

$$
\begin{aligned}
A(t+dt) &= A(t) - A(t)R_{A.}dt + C(t)R_{CA}dt + G(t)R_{GA}dt + T(t)R_{TA}dt \\
C(t+dt) &= C(t) + A(t)R_{AC}dt - C(t)R_{C.}dt + G(t)R_{GC}dt + T(t)R_{TC}dt \\
G(t+dt) &= G(t) + A(t)R_{AG}dt + C(t)R_{CG}dt - G(t)R_{G.}dt + T(t)R_{TG}dt \\
T(t+dt) &= T(t) + A(t)R_{AT}dt + C(t)R_{CT}dt + G(t)R_{GT}dt - T(t)R_{T.}dt
\end{aligned}
$$

where R_{XY} is the rate of instantaneous substitution from X to Y, and $R_{X.}$ Is the instantaneous rate of substitution from X to all other states.

Integrating over substitution histories along a branch

- Using matrix notation:

$$\begin{aligned}\mathbf{F}(t+dt) &= \mathbf{F}(t) + \mathbf{F}(t)\mathbf{R}dt \\ &= \mathbf{F}(t)(\mathbf{I} + \mathbf{R}dt)\end{aligned}$$

- Thus:

$$\frac{d\mathbf{F}(t)}{dt} = \mathbf{F}(t)\mathbf{R}$$

- Which can be solved as:

$$\begin{aligned}\mathbf{F}(t) &= \mathbf{F}(0)e^{\mathbf{R}t} \\ &= \mathbf{F}(0)\mathbf{P}(t)\end{aligned}$$

Discrete time interpretation

- Let’s compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$

Discrete time interpretation

- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$

Discrete time interpretation

- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$
  - If 1 substitution: $P_1(t) = F(0)\ Rt \rightarrow X=Rt$

Discrete time interpretation

- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$
  - If 1 substitution: $P_1(t) = F(0)\ Rt \rightarrow X=Rt$
  - If 2 substitutions: $P_2(t) = F(0)\ R^2*t^2 / 2! \rightarrow X=R^2*t^2 / 2!$

Discrete time interpretation

- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$
  - If 1 substitution: $P_1(t) = F(0)\ Rt \rightarrow X=Rt$
  - If 2 substitutions: $P_2(t) = F(0)\ R^2*t^2 / 2! \rightarrow X=R^2*t^2 / 2!$
  - If 3 substitutions: $P_3(t) = F(0)\ R^3*t^3 / 3! \rightarrow X=R^3*t^3 / 3!$

Discrete time interpretation

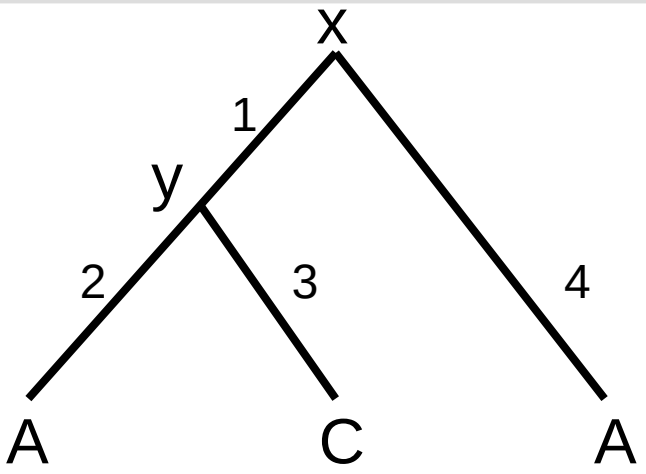
- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$
  - If 1 substitution: $P_1(t) = F(0)\ Rt \rightarrow X=Rt$
  - If 2 substitutions: $P_2(t) = F(0)\ R^2*t^2 / 2! \rightarrow X=R^2*t^2 / 2!$
  - If 3 substitutions: $P_3(t) = F(0)\ R^3*t^3 / 3! \rightarrow X=R^3*t^3 / 3!$
  - …

Discrete time interpretation

- Let's compute $P_n(t)$ with n the number of substitutions: $P_n(t) = F(0)*X$
  - If 0 substitution: $P_0(t) = F(0) \rightarrow X=Id$
  - If 1 substitution: $P_1(t) = F(0)\ Rt \rightarrow X=Rt$
  - If 2 substitutions: $P_2(t) = F(0)\ R^2*t^2 / 2! \rightarrow X=R^2*t^2 / 2!$
  - If 3 substitutions: $P_3(t) = F(0)\ R^3*t^3 / 3! \rightarrow X=R^3*t^3 / 3!$
  - …

→ *Can you see the link with the exponential function?*

Analytical computation of the likelihood: *P({A,C,A} | model)*

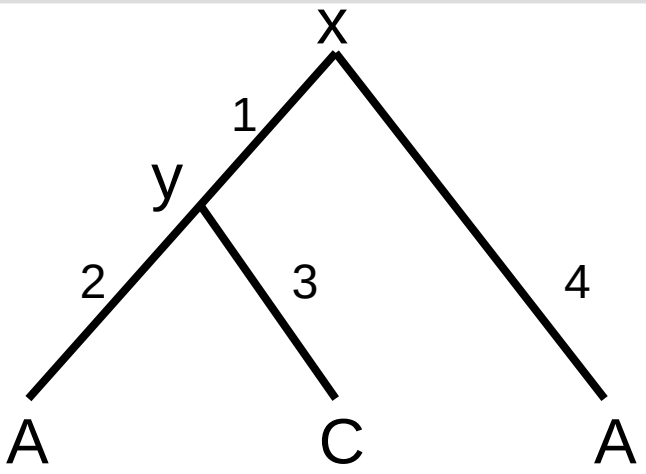


$$P(A,C,A,x,y|model) = \pi_x P_1(xy) P_2(yA) P_3(yC) P_4(xA)$$

With π_x probability to find base x at the root (in our simple case, ¼)

and $P_1(xy) = \exp(rt_1)$

Analytical computation of the likelihood: *P({A,C,A} | model)*



$$P(A,C,A,x,y|model) = \pi_x P_1(xy) P_2(yA) P_3(yC) P_4(xA)$$

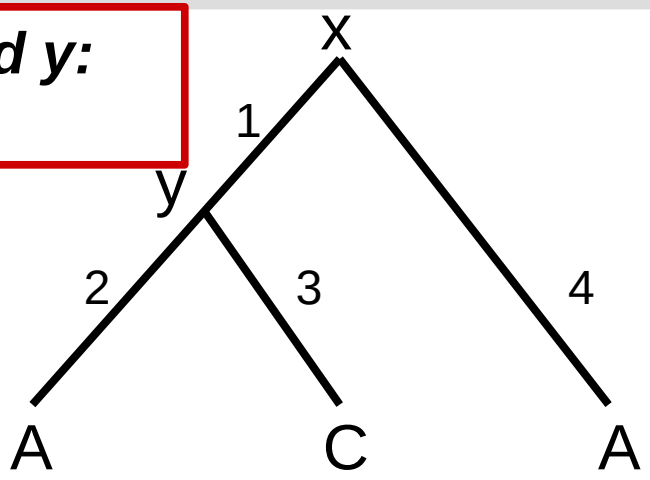
With π_x probability to find base x at the root (in our simple case, ¼)

and $P_1(xy) = \exp(rt_1)$

No need to simulate substitution histories over each branch!

Analytical computation of the likelihood: *P({A,C,A} | model)*

Integrating over x and y: see Alexis' talk!



$$P(A,C,A,x,y|model) = \pi_x P_1(xy) P_2(yA) P_3(yC) P_4(xA)$$

With π_x probability to find base x at the root (in our simple case, ¼)

and $P_1(xy) = \exp(rt_1)$

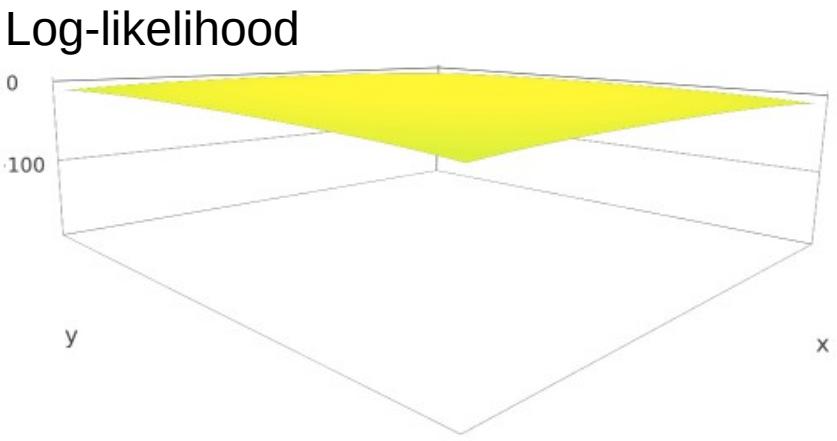
No need to simulate substitution histories over each branch!

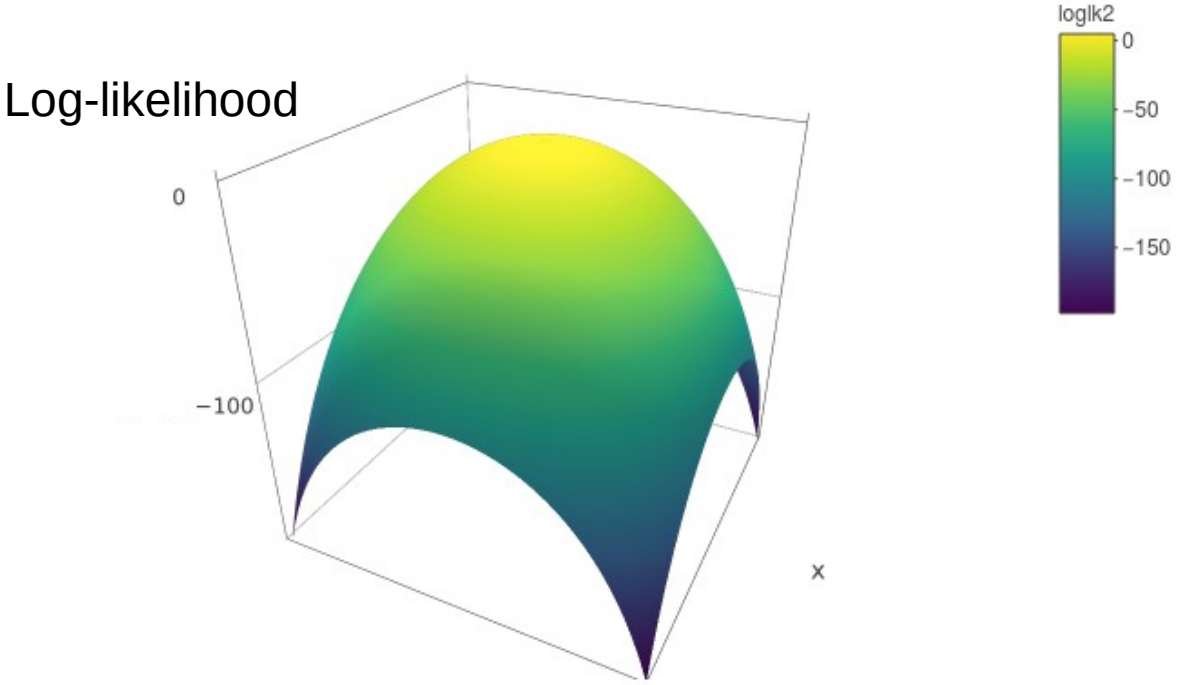
Plan: Criteria for evaluating phylogenies

- Criteria for evaluating phylogenetic trees:
  - Parsimony
  - Distance methods
  - Maximum Likelihood
  - Posterior probability (Bayesian approach)
- Conventions:
  - We’re dealing with aligned sequence data
  - gaps are not taken into account

Problems when relying on the maximum

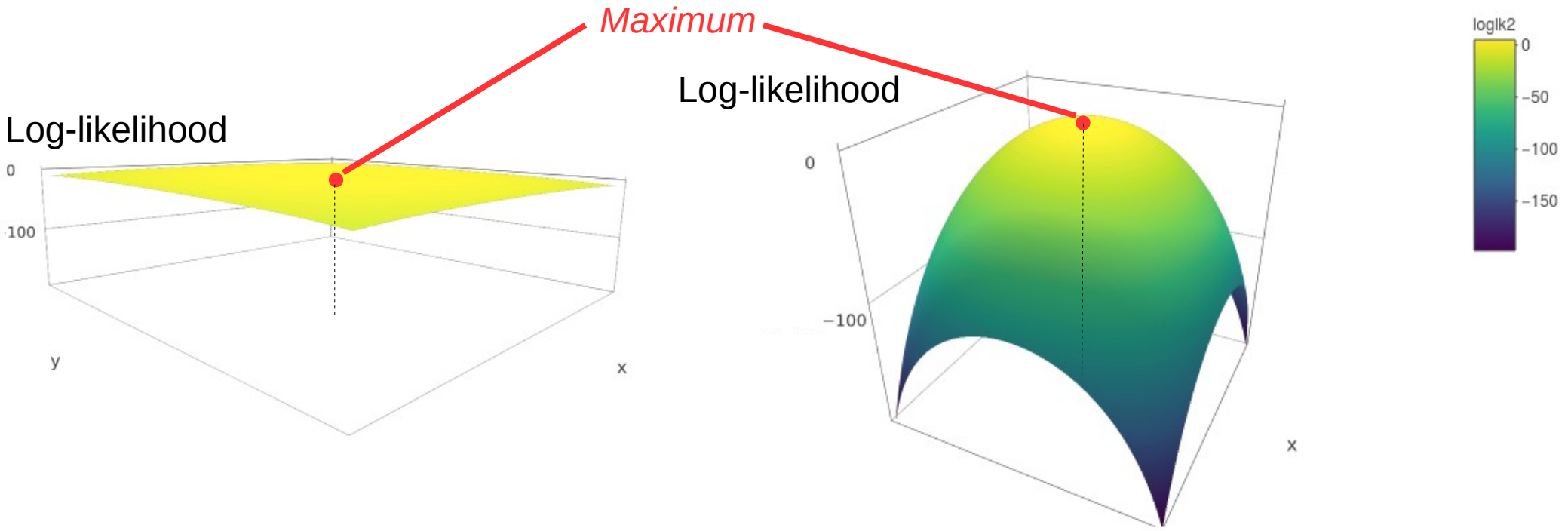
Ex. A: likelihood surface for a simple model with two parameters:





Problems when relying on the maximum

Ex. A: likelihood surface for a simple model with two parameters:



Problems when relying on the maximum

Ex. B : Given an alignment, *9* trees have very similar likelihoods and are much more likely than all the other ones.

Taking the most likely tree only provides knowledge about *1* of those *9* almost equi-likely trees.

→ It would be better to take into account all *9* of them!

Integrating instead of maximizing

- **Maximizing**: Looking for the topology and all other parameter values that are most likely
- **Integrating**: integrating over all topologies and parameter values according to their probability

  Pros :

  - If we are interested only in the topology, we integrate over all “nuisance” parameters
  - One gets confidence intervals for free

Likelihood vs Bayesian approach for sampling

Likelihood : $P(D|M,\theta)$

We want to sample parameter values θ of model M.

How can we know that the *9* sets of parameter values θ we have sampled are much more likely than all other θ, without sampling everything?

$\sum_{\theta} P(D|M,\theta) != 1 \rightarrow$ NOT a probability distribution.

Likelihood vs Bayesian approach for sampling

Likelihood : $P(D|M,\theta)$

We want to sample parameter values θ of model M.

How can we know that the *9* sets of parameter values θ we have sampled are much more likely than all other θ, without sampling everything?

$\sum_{\theta} P(D|M,\theta) != 1 \rightarrow$ NOT a probability distribution.

But :

$\sum_{\theta} P(\theta|D,M) = 1 \rightarrow$ True probability distribution.

If the added probabilities of *9* sets of parameter values θ reach *0.99*, I know I have sampled all the most probable parameter values, without needing to sample more!

Likelihood vs Bayesian approach for sampling

Likelihood : $P(D|M,\theta)$

We want to sample parameter values θ of model M.

How can we know that the *9* sets of parameter values θ we have sampled are much more likely than all other θ, without sampling everything?

$\sum_{\theta} P(D|M,\theta) != 1 \rightarrow$ NOT a probability distribution.

But :

$\sum_{\theta} P(\theta|D,M) = 1 \rightarrow$ True probability distribution.

Posterior probability

If the added probabilities of *9* sets of parameter values θ reach *0.99*, I know I have sampled all the most probable parameter values, without needing to sample more!

Simplifying the notation

When we write $P(\theta|D,M)$, we mean that we are computing the probability of parameter values θ given model M and given data D.

Here M is used to represent the structure of the model.

For instance:

- The fact that all sites are independent
- The fact that they share a single value of the transition/transversion ratio
- The fact that we have a Birth-death prior on a chronogram
- Etc…

In some cases, we may want to integrate over different model structures M: e.g. a model with 1 value of the transition/transversion ratio, or 2, or 3…

In most cases, we use a single model M, so we will forget about it in most of the following:

$$P(\theta|D,M) \rightarrow P(\theta|D)$$

Bayesian inference

Bayes theorem:

$$P(\theta|D) = \frac{P(\theta \wedge D)}{P(D)} = \frac{P(D \wedge \theta)}{P(D)}$$

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

Bayesian inference

Bayes theorem:

$$P(\theta|D) = \frac{P(\theta \wedge D)}{P(D)} = \frac{P(D \wedge \theta)}{P(D)}$$

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{\int_{\theta} P(D|\theta)\,d\theta}$$

Bayesian inference

Bayes theorem:

$$P(\theta|D) = \frac{P(\theta \wedge D)}{P(D)} = \frac{P(D \wedge \theta)}{P(D)}$$

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)}$$

The importance of prior probabilities

The Bayesian approach amounts to considering that all parameters of a model are random variables in a probabilistic world.

Therefore, one needs to assign probability distributions to those parameters: the **priors**.

$$P(\theta|D) = P(D|\theta)\,P(\theta)/P(D)$$

→ PROS: allows incorporating prior information coming from the analysis of other data: my conclusion relies on more than the tiny amount of data I have analyzed in a given experiment.

→ CONS: introduces prior information into the analysis: is my conclusion simply propagating my prior information?

Bayesian inference

Bayes theorem: $$P(\theta|D)=\frac{P(D|\theta)P(\theta)}{P(D)}$$

We are interested in the posterior probability of the parameter values: we want to **sample parameter values according to their posterior probability given the data**.
Therefore we need to get the distribution for *P(θ|D)*.

We have two solutions:

- *Integrate over all parameter values*
  - Requires some maths
  - Not always possible
- *Sample from this distribution in some (smart) way*

Naive sampling of *P(θ|D)*

Random sampling

Can work for small problems.

- Inference of the probability of getting heads from a coin

http://rpubs.com/boussau/384012

Naive sampling of *P(θ|D)*

Random sampling

Can work for small problems.

- Inference of the probability of getting heads from a coin:

http://rpubs.com/boussau/384012

Problem : in phylogenetics, for *20* sequences, there are already *221. 10^{18}* possible topologies… for which one would want to sample branch lengths and other parameter values.

→ We cannot reasonably hope to sample trees with a good probability using random sampling

Naive sampling of *P(θ|D)*

Random sampling

Can work for small problems.

- Inference of the probability of getting heads from a coin:

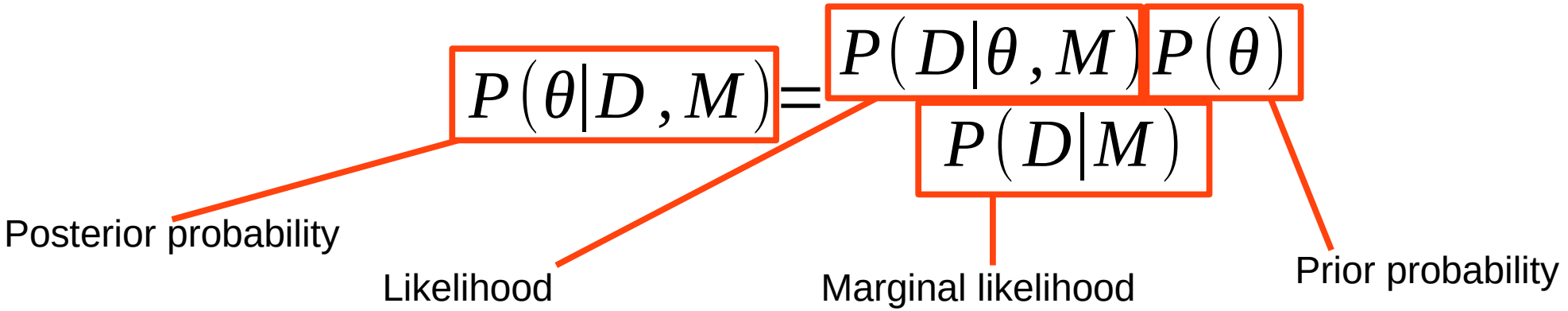
http://rpubs.com/boussau/384012

Will not work for complex problems : in phylogenetics, for *20* sequences, there are already *221. 10^{18}* possible topologies… for which one would want to sample branch lengths and other parameter values.

→ **We cannot reasonably hope to sample trees with a good probability using random sampling**

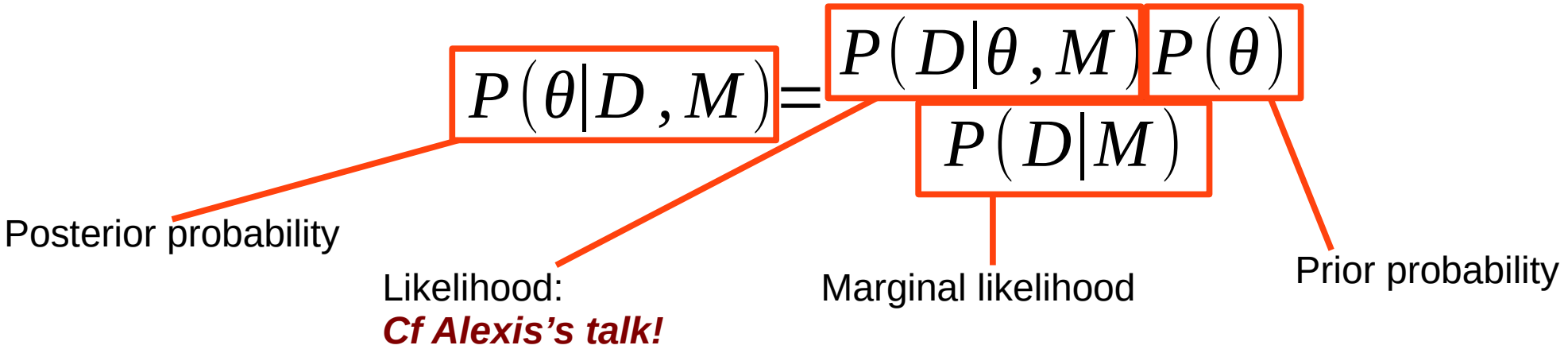
Tomorrow we will see smarter ways to sample from complex posterior distributions

Bayesian phylogenetics



- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Bayesian phylogenetics



- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Bayesian phylogenetics

$$P(\theta|D,M) = \frac{P(D|\theta,M)\,P(\theta)}{P(D|M)}$$

Posterior probability

Likelihood:
Cf Alexis's talk!

Marginal likelihood:
In lots of cases, it's OK not to compute it

Prior probability

- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Bayesian phylogenetics

$$P(\theta|D,M) = \frac{P(D|\theta,M)P(\theta)}{P(D|M)}$$

Posterior probability

Likelihood:
Cf Alexis's talk!

Marginal likelihood:
In lots of cases, it's OK not to compute it

Prior probability:
We need prior distributions for all parameters

- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Bayesian phylogenetics

In phylogenetics, one may want to sample:

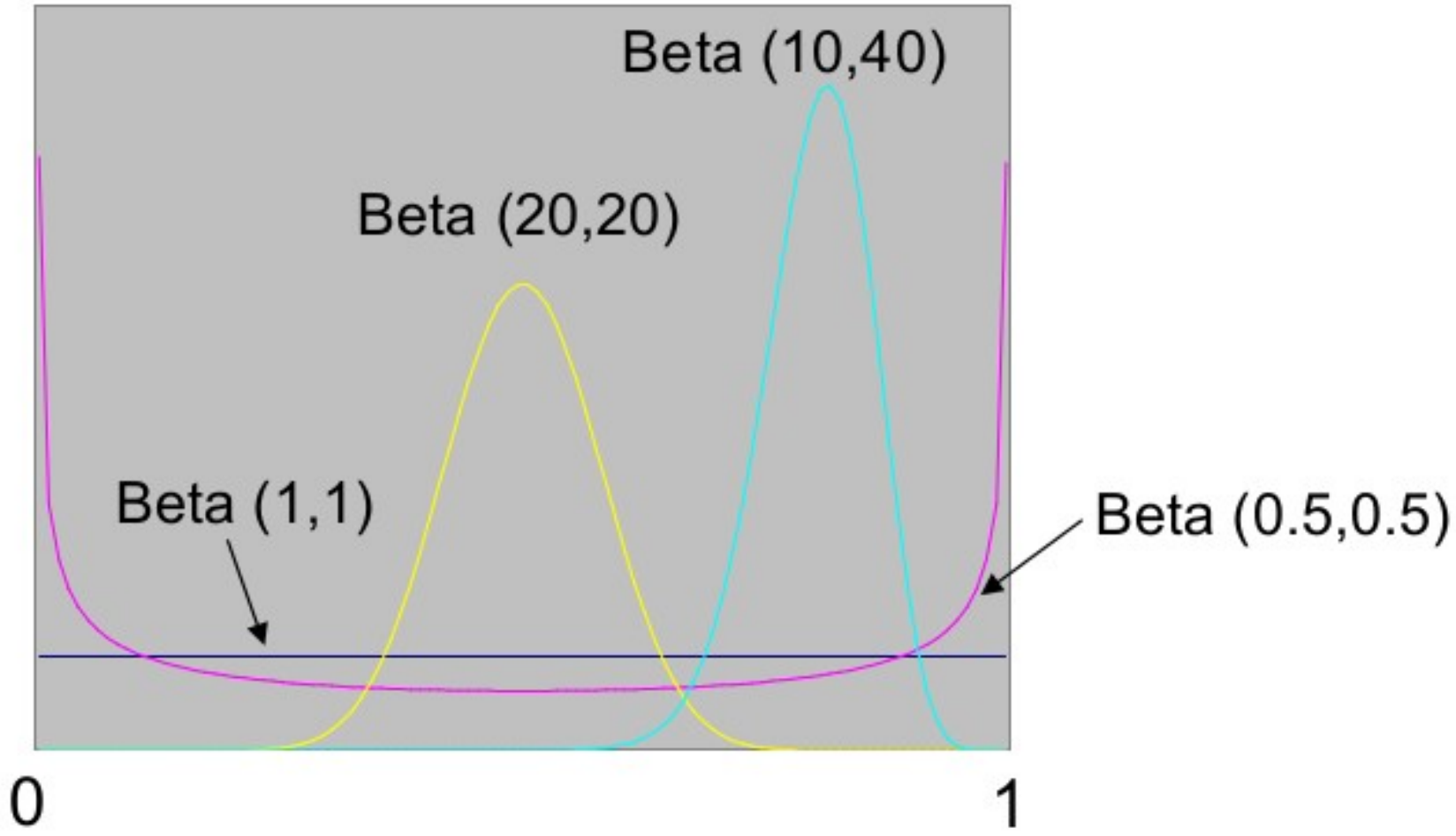
- Phylogenetic trees (when we don’t care about dates):
  - topologies
  - branch lengths
- Chronograms (when dates are of interest)
- Rates:
  - Rates of exchangeability
  - Site-wise rates of evolution
  - Birth-death rates
  - ...
- Frequencies:
    - ACGT equilibrium frequencies
    - Root frequencies
- Other parameters...

MCMC in phylogenetics

Parameter	Prior (example)
Topology	Uniform...
Branch lengths	Exponential, Gamma+Dirichlet...
Chronogram	Birth-Death, Coalescent...
Rates of exchangeability	Dirichlet...
Site-wise rates of evolution	(Discretized) Gamma...
Birth-death rates	Lognormal, Exponential...
ACGT equilibrium frequencies	Dirichlet...
Root frequencies	Dirichlet...

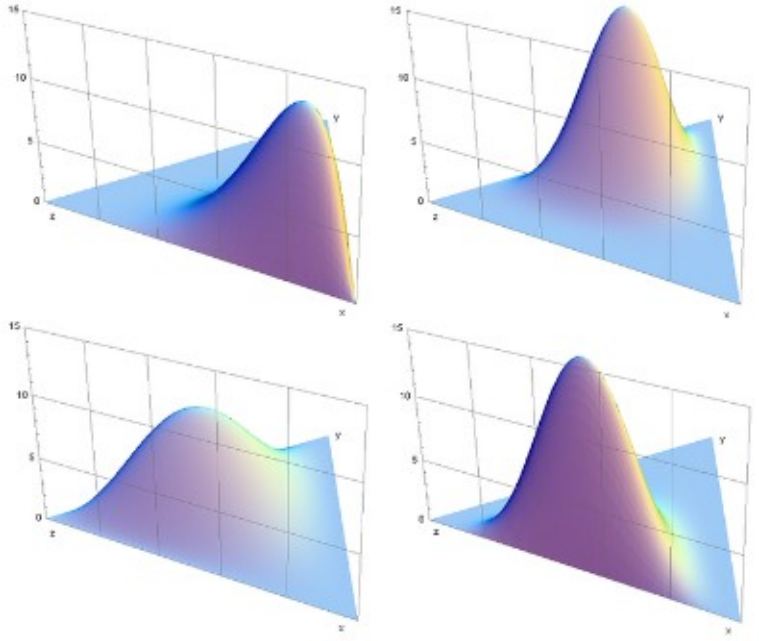
Prior for proportions

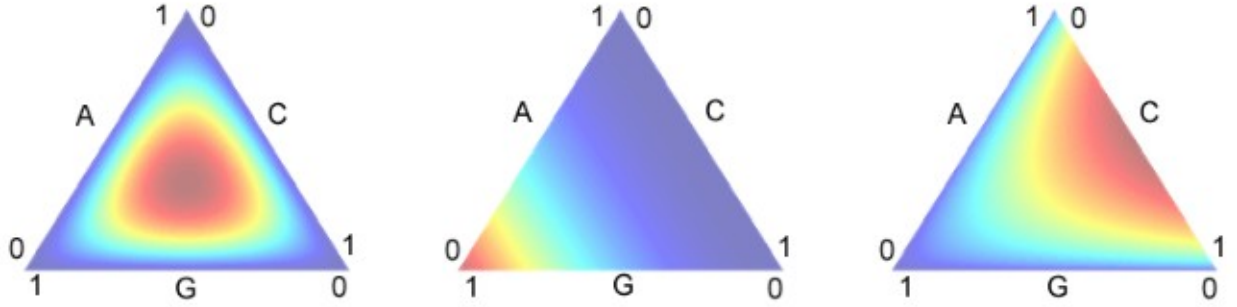
- Beta prior



Prior for simplices (ACGT equilibrium frequencies, exchangeability rates...)

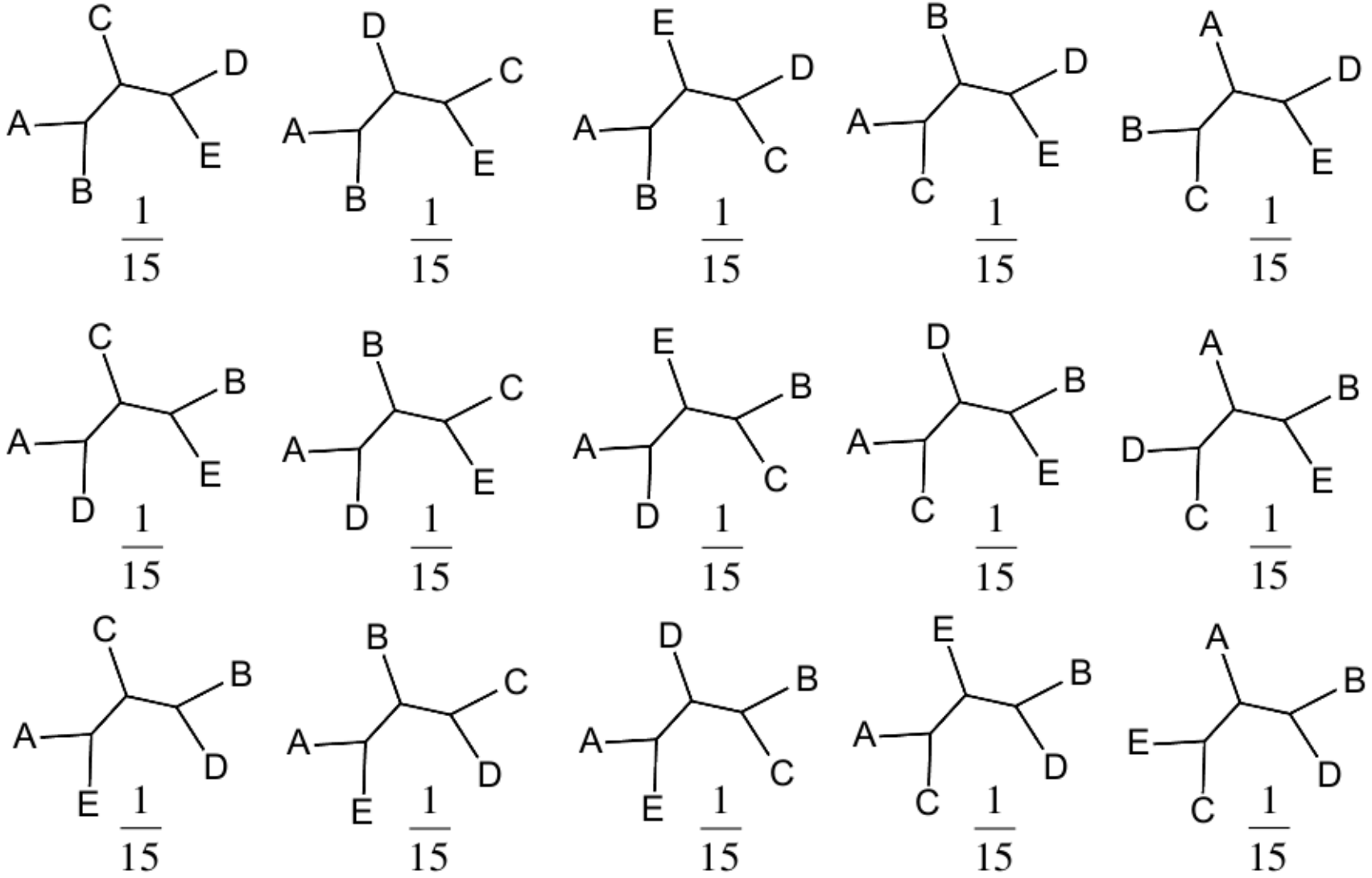
- Dirichlet prior (*i.e.* Beta but in more dimensions)





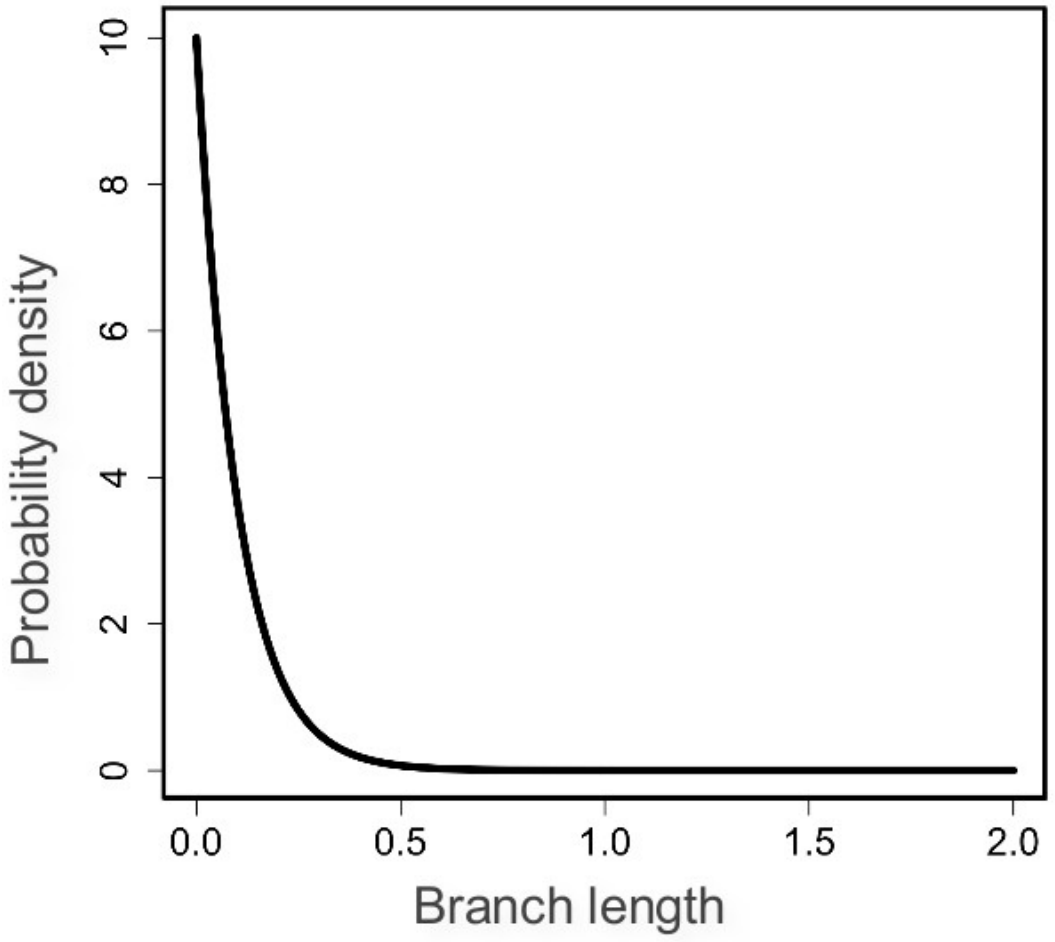
Unrooted topology prior

- Discrete uniform prior



Prior for branch lengths

- Exponential prior (λ=10, mean=0.1)

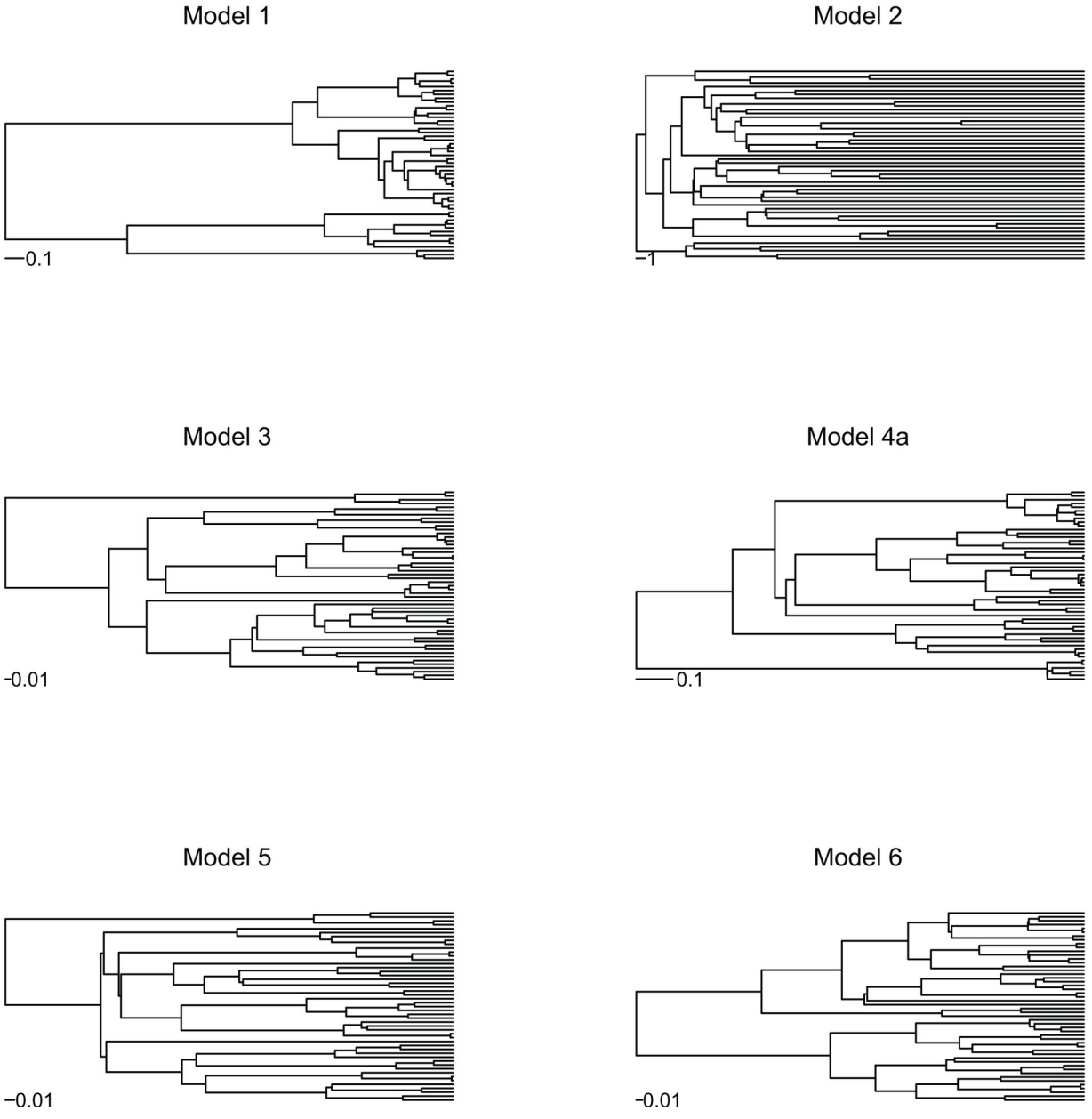


- Alternative: Gamma for total tree length+Dirichlet

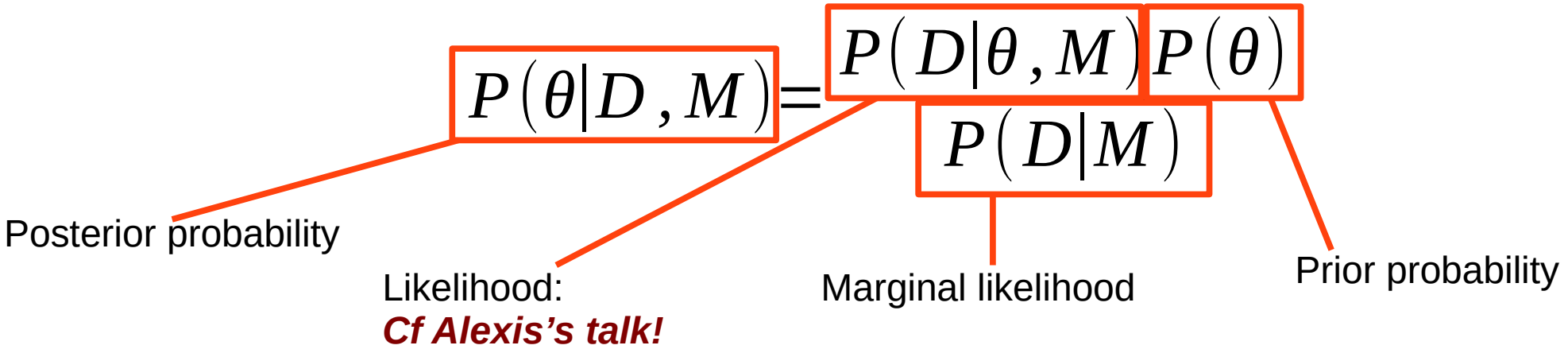
Chronogram prior: Birth-death process

Phylogenies simulated under a model with saturated diversity and a constant turnover rate (Model 1) have short terminal branches compared to phylogenies simulated under the pure-birth process (Yule model; Model 5) With saturated diversity but decaying turnover rates, terminal branches become longer (Model 2). Compared to the pure-birth process (Model 5), the presence of extinction pushes phylogenetic nodes towards the tips (Model 3), whereas a decay in speciation rate pushes them towards the root (Model 6) In the presence of both extinction and a decay in speciation rate (Model 4), however, these two effects counteract, producing a phylogeny that appears similar to the pure-birth model. All phylogenies were simulated with the same initial speciation rate (six speciation events per time unit). The extinction rate in Models 3 and 4a was identical (three speciation events per time unit). The exponential variation in speciation rate in Models 2, 4a, and 6 was identical (0.25 per time unit). Note the different time scales.

Morlon et al., 2010



Bayesian phylogenetics



- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Bayesian phylogenetics

$$P(\theta|D,M) = \frac{P(D|\theta,M)\,P(\theta)}{P(D|M)}$$

Posterior probability

Likelihood:
Cf Alexis's talk!

Marginal likelihood

Prior probability

$$P(\theta) = P(topology)\,P(branch\,lengths)\,P(exchangeabilities)\,P(equilibrium\,frequencies)\ldots$$

- In phylogenetics:
  - D: aligned sequence data
  - M: the model can be very complicated, but usually contains:
    - Topology
    - Branch lengths
    - Rate matrix
    - Etc…
  - θ: the values of the parameters above

Smart sampling of *P(θ|D)*

- *Smart sampling = no need to sample a huge number of points !*

→ Ideally : Sample trees and parameter values with a frequency equal to their probability.

E.g.: we would sample one of our *9* trees *99%* of the time!

Several approaches:

- *Importance sampling*
- *Markov Chain Monte Carlo (MCMC)*
- Sequential Monte Carlo
- ...

Smart sampling of *P(θ|D)*

- *Smart sampling = no need to sample a huge number of points !*

→ Ideally : Sample trees and parameter values with a frequency equal to their probability.

E.g.: we would sample one of our *9* trees *99%* of the time!

Several approaches:

- *Importance sampling*
- *Markov Chain Monte Carlo (MCMC)*
- Sequential Monte Carlo
- ...

Next time!

Conclusion

- One can perform inference according to the posterior probability of a probabilistic model
- That’s what Bayesian inference is about
- It combines the likelihood with priors on parameter values
- In phylogenetics, the likelihood is typically computed thanks to Felsenstein’s pruning algorithm (1981, cf Alexis)
- Then priors need to be defined for:
  - Topologies and branch lengths / chronograms
  - Rates
  - Other parameters…
- When parameters are independent, to compute the prior of all parameter values, one only needs to compute the product over individual parameter priors

Plan: Criteria for evaluating phylogenies

- Criteria for evaluating phylogenetic trees:
  - Parsimony
  - Distance methods
  - Maximum Likelihood
  - Posterior probability (Bayesian approach)
- Conventions:
  - We’re dealing with aligned sequence data
  - gaps are not taken into account

Optimization algorithms (Alexis, afternoon)

MCMC (Mike, tomorrow)