Phylogenetic Inference with Markov Chain Monte Carlo

Franco Aquistapace Tagua

Information Theory and Inference

Department of Physics and Astronomy
University of Padua

Goals

- **Simulate** phylogenetic data → phylogenetic trees of aligned sequences
- Perform **inference** on simulated data → reconstruct the tree from the sequences

Goals

- **Simulate** phylogenetic data → phylogenetic trees of aligned sequences
- Perform **inference** on simulated data → reconstruct the tree from the sequences

Outline

- **Introduction** → What is phylogenetics? What are the main things to consider?
- **Development** → How can we simulate and infer phylogenetic trees?
- **Results** → What can we say about our simulated / inferred trees?
- **Conclusion** → Did we achieve our goals?

INTRODUCTION

The problem of phylogenetics

What is **phylogenetics**?

- Sequences from different organisms → infer a tree that describes their **evolutionary history**
  - DNA, proteins, morphology
  - Aligned

The problem of phylogenetics

What is **phylogenetics**?

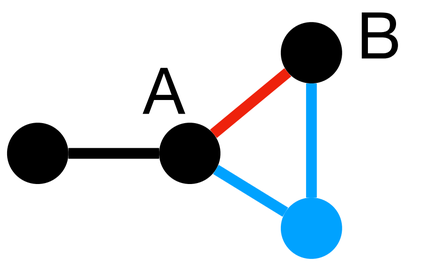
- Sequences from different organisms → infer a tree that describes their **evolutionary history**
  - DNA, proteins, morphology
  - Aligned

- Inference → deterministic / probabilistic approaches
  - Definition of sequence *similarity*
  - Model of sequence evolution

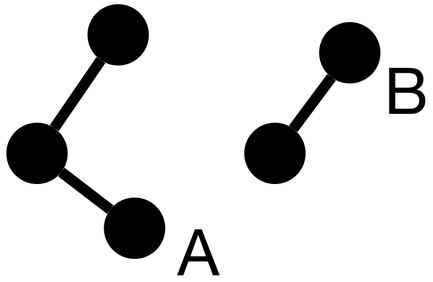
The problem of phylogenetics

What is a **tree**?

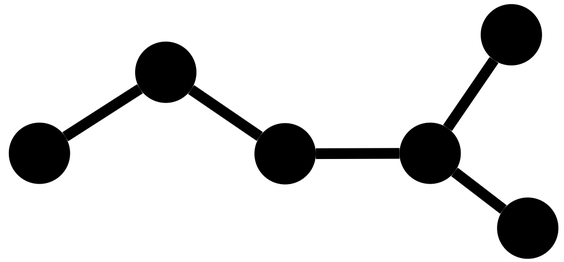
Connected graph with no *cycles* → any two nodes are connected by a single path



Not a tree



Not a tree

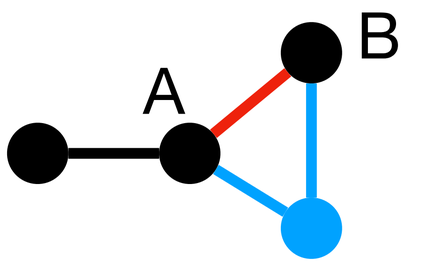


Tree

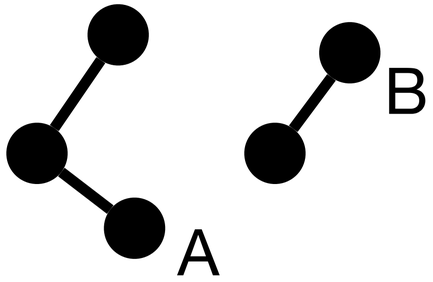
The problem of phylogenetics

What is a **tree**?

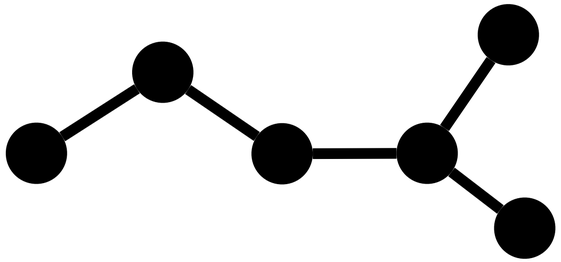
Connected graph with no *cycles* → any two nodes are connected by a single path



Not a tree

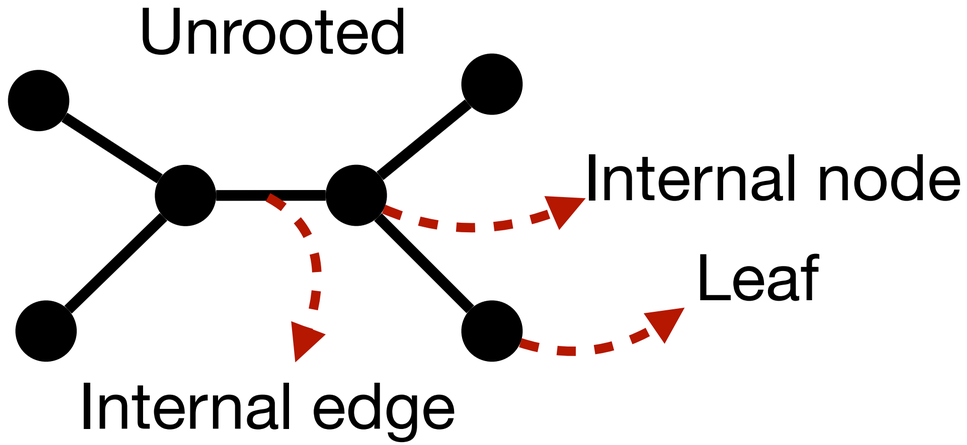


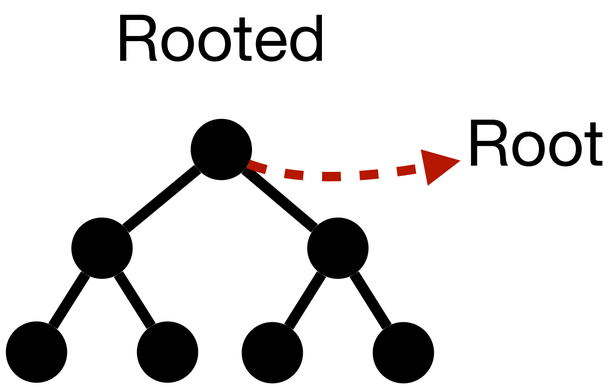
Not a tree



Tree

Binary tree → every internal node has degree 3 (except maybe for the *root*)





Probabilistic models of DNA mutation

Evolution of sequence sites → conditional probabilities

$$\left.\begin{array}{l} S_0 : R\ R\ Y\ R\ Y\ R\ \ \ldots \\ S_1 : R\ R\ Y\ R\ Y\ Y\ \ \ldots \end{array}\right\} P(S_1 = x \,|\, S_0 = y)$$

Probabilistic models of DNA mutation

Evolution of sequence sites → conditional probabilities

$$\left.\begin{array}{l} S_0 : R\ R\ Y\ R\ Y\ R\ \ldots \\ S_1 : R\ R\ Y\ R\ Y\ Y\ \ldots \end{array}\right\} P(S_1 = x \,|\, S_0 = y)$$

We need:

- Base probability distribution → $\overrightarrow{p_0}$
- Rate matrix → Q
- Transition matrix → $M(t) = \exp(Qt)$

Probabilistic models of DNA mutation

Evolution of sequence sites → conditional probabilities

$$\left.\begin{array}{l} S_0 : R\ R\ Y\ R\ Y\ R\ \ldots \\ S_1 : R\ R\ Y\ R\ Y\ Y\ \ldots \end{array}\right\} P(S_1 = x \,|\, S_0 = y)$$

We need:

- Base probability distribution → $\overrightarrow{p_0}$
- Rate matrix → Q
- Transition matrix → $M(t) = \exp(Qt)$

➡ $\overrightarrow{p_t} = \overrightarrow{p_0} M(t)$

Probabilistic models of DNA mutation

Evolution of sequence sites → conditional probabilities

$$\left.\begin{array}{l} S_0 : R\ R\ Y\ R\ Y\ R\ \ldots \\ S_1 : R\ R\ Y\ R\ Y\ Y\ \ldots \end{array}\right\} P(S_1 = x \,|\, S_0 = y)$$

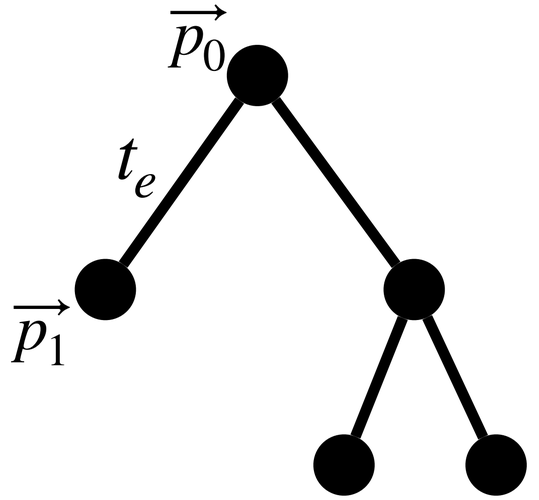
We need:

- Base probability distribution → $\overrightarrow{p_0}$
- Rate matrix → Q
- Transition matrix → $M(t) = \exp(Qt)$

➡ $\overrightarrow{p_t} = \overrightarrow{p_0} M(t)$

Only mechanism considered is substitution, others are excluded (deletion, insertion, …)

Application to a tree



$$\overrightarrow{p_1} = \overrightarrow{p_0} M(t_e)$$

The Jukes-Cantor model

Simplest Markov model of base substitution

- Uniform root distribution $\rightarrow \left(\overrightarrow{p_\rho}\right)_i = |\chi|^{-1}, \forall i$
- Uniform transition rates $\rightarrow Q_{ij} = -\alpha\delta_{ij} + (\alpha/3)(1-\delta_{ij})$

$|\chi|$: Size of the alphabet

The Jukes-Cantor model

Simplest Markov model of base substitution

- Uniform root distribution $\rightarrow \left(\overrightarrow{p_\rho}\right)_i = |\chi|^{-1}, \forall i$ $|\chi|$: Size of the alphabet
- Uniform transition rates $\rightarrow Q_{ij} = -\alpha\delta_{ij} + (\alpha/3)(1-\delta_{ij})$

➡ $M(t)_{ij} = (1-a(t))\delta_{ij} + (a(t)/3)(1-\delta_{ij})$ with $a(t) = \frac{3}{4}\left[1 - \exp\left(-frac43\alpha t\right)\right]$

The Jukes-Cantor model

Simplest Markov model of base substitution

- Uniform root distribution $\rightarrow \left(\overrightarrow{p_\rho}\right)_i = |\chi|^{-1}, \forall i$ $\quad |\chi|$: Size of the alphabet
- Uniform transition rates $\rightarrow Q_{ij} = -\alpha\delta_{ij} + (\alpha/3)(1-\delta_{ij})$

➡ $M(t)_{ij} = (1-a(t))\delta_{ij} + (a(t)/3)(1-\delta_{ij})$ with $a(t) = \frac{3}{4}\left[1 - \exp\left(-frac43\alpha t\right)\right]$

Since $\overrightarrow{p_\rho}M(t) = \overrightarrow{p_\rho}, \forall t$ $\rightarrow$ the base distribution remains uniform at every node.

The Markov Chain Monte Carlo approach

MCMC for tree inference → **Metropolis—Hastings** algorithm

1. Set of aligned sequences (S)

The Markov Chain Monte Carlo approach

MCMC for tree inference → **Metropolis—Hastings** algorithm

1. Set of aligned sequences (S)
2. Initialize tree topology (T) and parameters ($\{\theta\}$)

The Markov Chain Monte Carlo approach

MCMC for tree inference → **Metropolis—Hastings** algorithm

1. Set of aligned sequences (S)
2. Initialize tree topology (T) and parameters ($\{\theta\}$)
3. Propose new topology + parameters (T', $\{\theta'\}$) → symmetric probabilistic process

The Markov Chain Monte Carlo approach

MCMC for tree inference → **Metropolis—Hastings** algorithm

1. Set of aligned sequences (S)
2. Initialize tree topology (T) and parameters $(\{\theta\})$
3. Propose new topology + parameters $(T', \{\theta'\}) \rightarrow$ symmetric probabilistic process
4. Accept proposal with probability:

$$A = \min\left[1, \underbrace{\frac{P(S \mid T', \{\theta'\})}{P(S \mid T, \{\theta\})}}_{\text{Likelihood}} \cdot \underbrace{\frac{P(T', \{\theta'\})}{P(T, \{\theta\})}}_{\text{Prior}}\right]$$

The Markov Chain Monte Carlo approach

MCMC for tree inference → **Metropolis—Hastings** algorithm

1. Set of aligned sequences (S)
2. Initialize tree topology (T) and parameters $(\{\theta\})$
3. Propose new topology + parameters $(T', \{\theta'\}) \rightarrow$ symmetric probabilistic process
4. Accept proposal with probability:

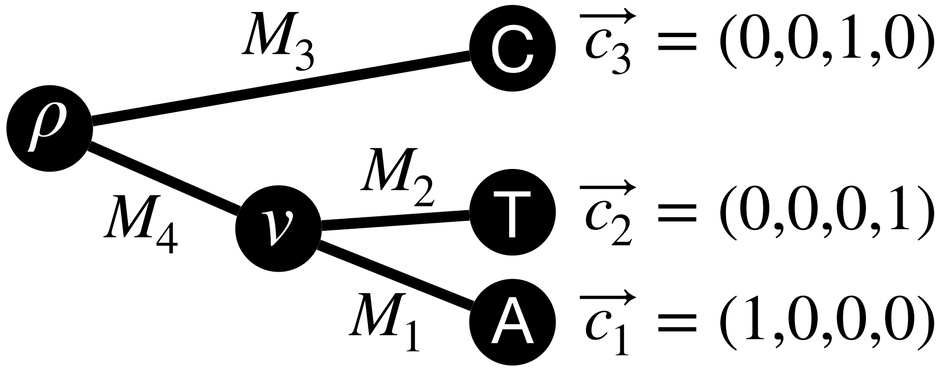
$$A = \min\left[1, \underbrace{\frac{P(S \mid T', \{\theta'\})}{P(S \mid T, \{\theta\})}}_{\text{Likelihood}} \cdot \underbrace{\frac{P(T', \{\theta'\})}{P(T, \{\theta\})}}_{\text{Prior}}\right]$$

5. Repeat steps 3-4 N_{MC} times to generate samples

The Markov Chain Monte Carlo approach

How can we calculate the likelihood? → **Felsenstein's pruning algorithm**

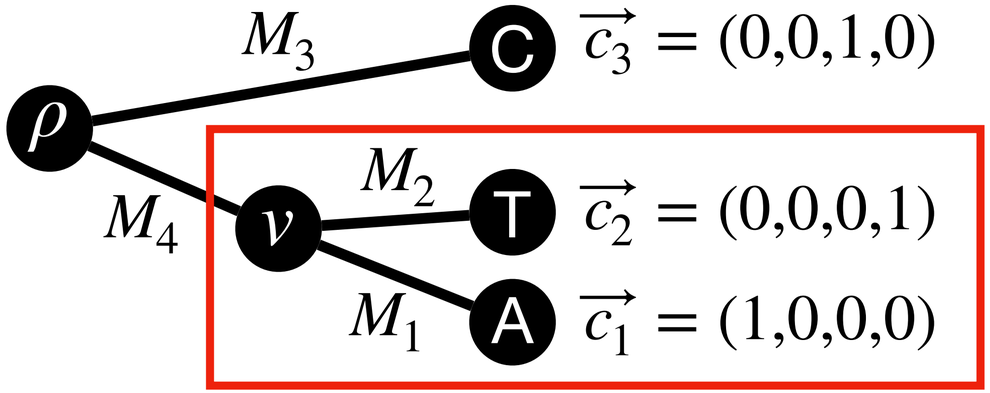
1.



The Markov Chain Monte Carlo approach

How can we calculate the likelihood? → **Felsenstein's pruning algorithm**

1.

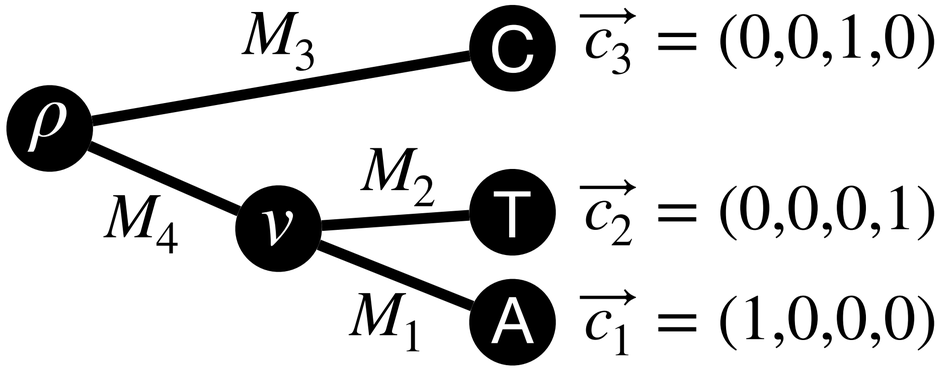


$\vec{c}_v = \overrightarrow{w_1} \odot \overrightarrow{w_2}$ with $\overrightarrow{w_n} = M_n \overrightarrow{c_n}^T$

The Markov Chain Monte Carlo approach

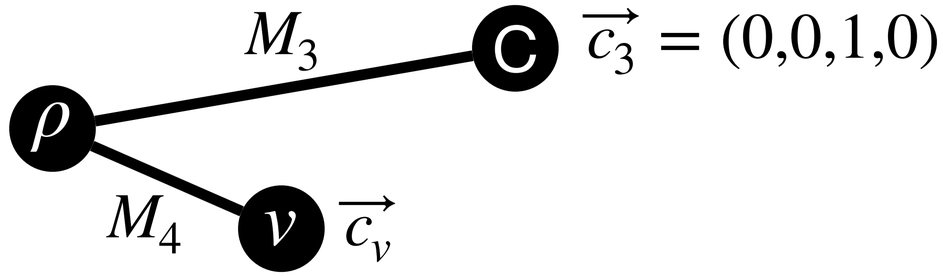
How can we calculate the likelihood? → **Felsenstein's pruning algorithm**

1.



$$\vec{c}_v = \overrightarrow{w_1} \odot \overrightarrow{w_2} \quad \text{with} \quad \overrightarrow{w_n} = M_n \overrightarrow{c_n}^T$$

2.

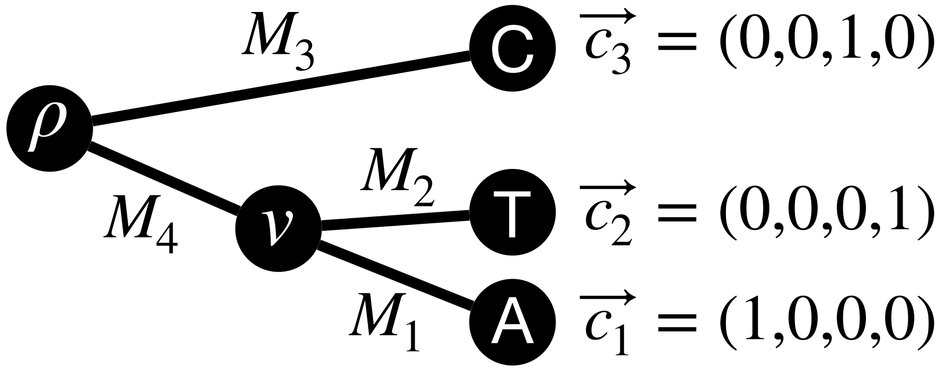


$$\vec{c}_\rho = \overrightarrow{w_v} \odot \overrightarrow{w_3}$$

The Markov Chain Monte Carlo approach

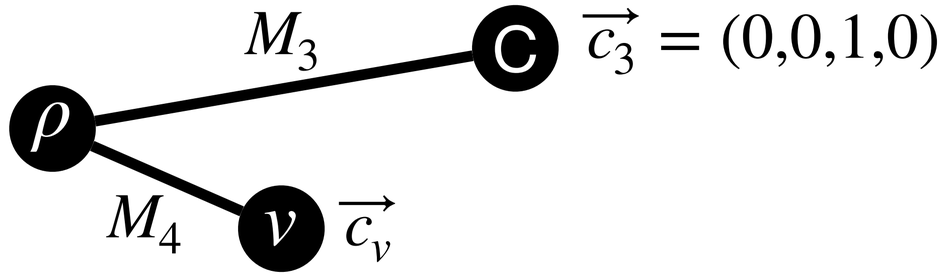
How can we calculate the likelihood? → **Felsenstein's pruning algorithm**

1.



$$\vec{c}_v = \overrightarrow{w_1} \odot \overrightarrow{w_2} \quad \text{with} \quad \overrightarrow{w_n} = M_n \overrightarrow{c_n}^T$$

2.



$$\vec{c}_\rho = \overrightarrow{w_v} \odot \overrightarrow{w_3}$$

3. $P(S \mid T, \{\theta\}) = \vec{p}_\rho \cdot \vec{c}_\rho$

Tree space

For a fixed collection X of n taxa $\rightarrow$ space of distinct topological trees

- Unrooted binary trees $\rightarrow (2n-5)!!$
- Rooted binary trees $\rightarrow (2n-3)!!$

Tree space

For a fixed collection X of n taxa $\rightarrow$ space of distinct topological trees

- Unrooted binary trees $\rightarrow (2n-5)!!$ $\quad \Rightarrow \quad n = 10 \rightarrow \sim 2$ million trees
- Rooted binary trees $\rightarrow (2n-3)!!$

Tree space

For a fixed collection X of n taxa $\rightarrow$ space of distinct topological trees

- Unrooted binary trees $\rightarrow (2n-5)!!$ $\quad \Rightarrow \quad n = 10 \rightarrow \sim 2$ million trees
- Rooted binary trees $\rightarrow (2n-3)!!$

How to **move** around tree space?

- Nearest—neighbor interchange (NNI)
- Subtree prune and regraft (SPR)
- Tree bisection and reconnection (TBR)

Tree space

For a fixed collection X of n taxa $\rightarrow$ space of distinct topological trees

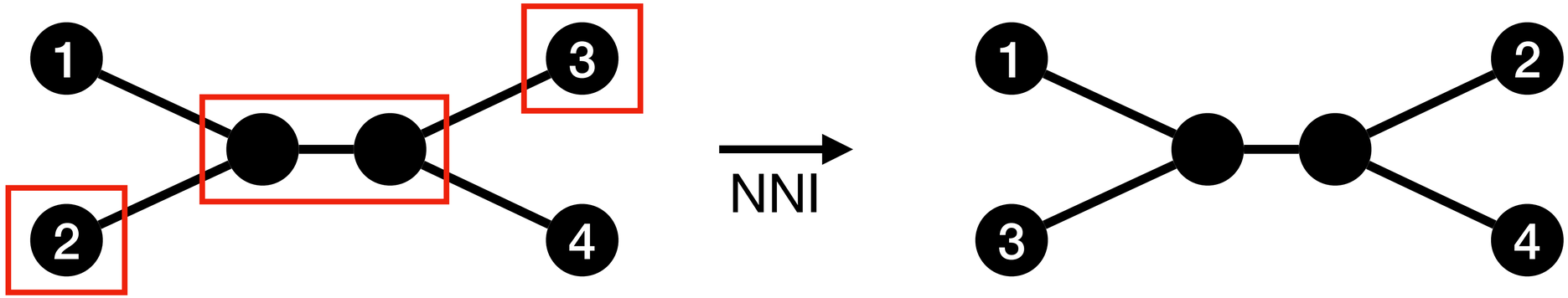
- Unrooted binary trees $\rightarrow (2n-5)!!$ $\quad \Rightarrow \quad n = 10 \rightarrow \sim 2$ million trees
- Rooted binary trees $\rightarrow (2n-3)!!$

How to **move** around tree space?

- Nearest—neighbor interchange (NNI)
- Subtree prune and regraft (SPR)
- Tree bisection and reconnection (TBR)

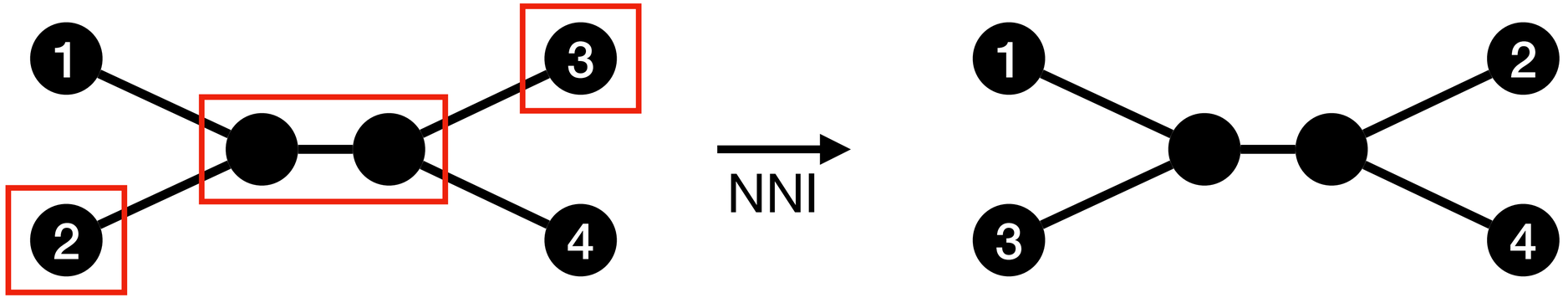
Tree space

Nearest—neighbor interchange (NNI)

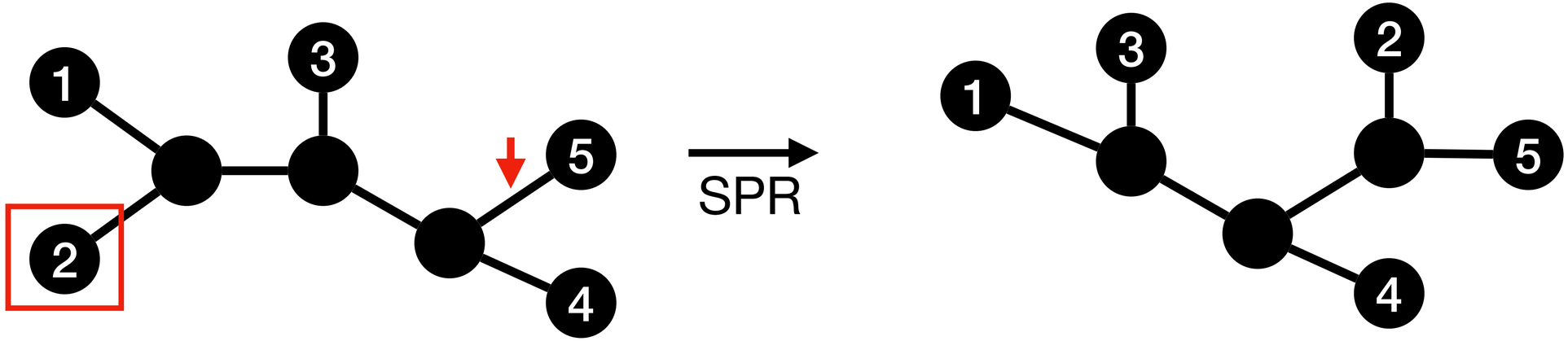


Tree space

Nearest—neighbor interchange (NNI)



Subtree prune and regraft (SPR)



DEVELOPMENT

Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

Can we sample the tree space uniformly?

- Only way I could think of was to build all possible trees and choose one uniformly at random

  ➡ Very inefficient

Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

Can we sample the tree space uniformly?

- Only way I could think of was to build all possible trees and choose one uniformly at random

  ➡ Very inefficient

What about a *constructive* approach?

- Need to be careful → growing uniformly is biased by isomorphic configurations

Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

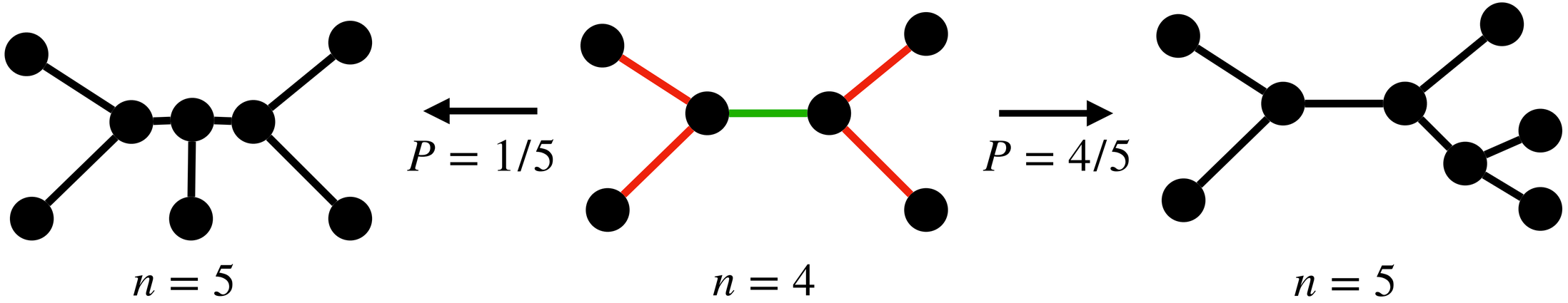
Can we sample the tree space uniformly?

- Only way I could think of was to build all possible trees and choose one uniformly at random

  ➡ Very inefficient

What about a *constructive* approach?

- Need to be careful → growing uniformly is biased by isomorphic configurations



Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

What about a constructive approach that removes isomorphisms?

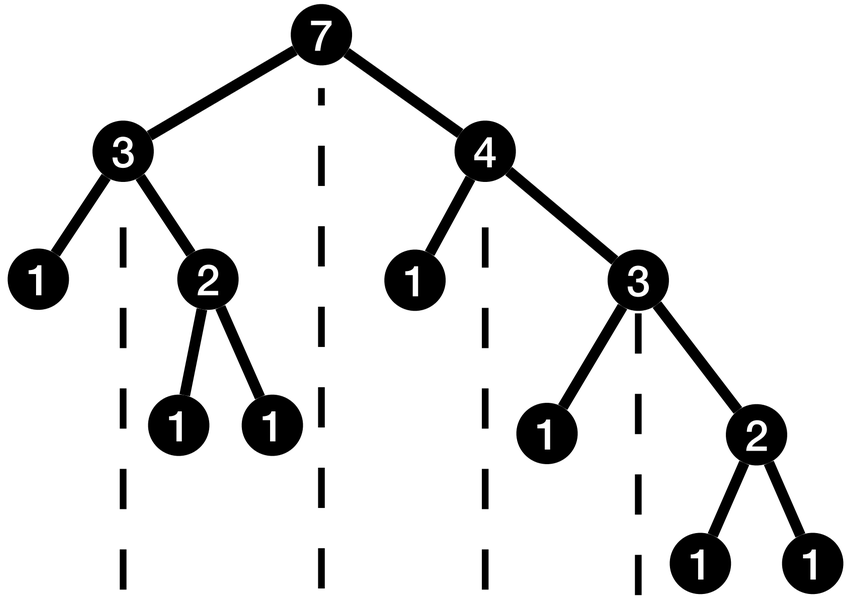
- **Constraint:** any subtree in a tree must be organized so that the main branch with most leaves is on the right

Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

What about a constructive approach that removes isomorphisms?

- **Constraint:** any subtree in a tree must be organized so that the main branch with most leaves is on the right

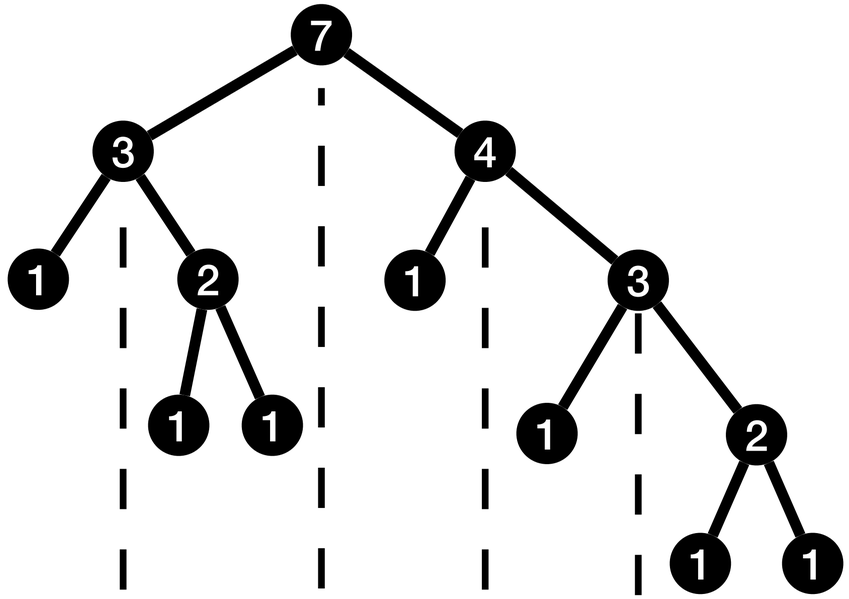


Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

What about a constructive approach that removes isomorphisms?

- **Constraint:** any subtree in a tree must be organized so that the main branch with most leaves is on the right



$$7$$

$$3 \,|\, 4$$

$$1 \,|\, 2 \,\big|\, 1 \,|\, 3$$

$$1 \,\big|\, 1 \,|\, 1 \,\big|\, 1 \,\big|\, 1 \,|\, 2$$

$$\cdots$$

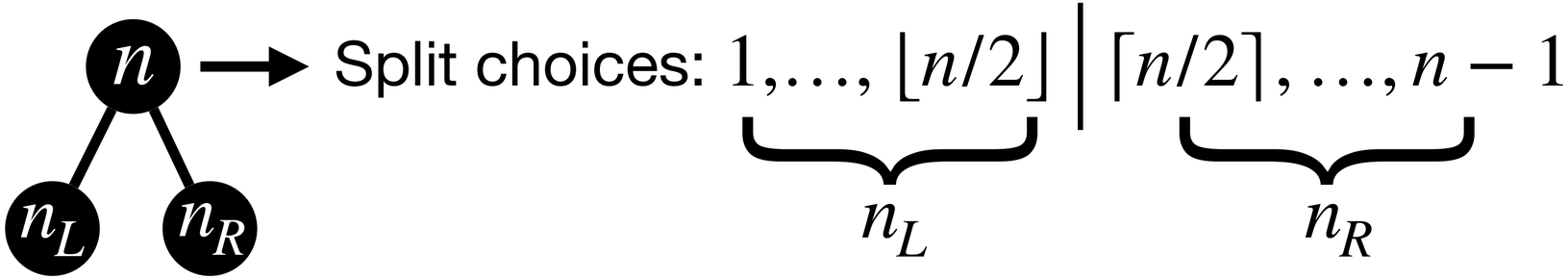
Largest *trivial* representation

Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

What about a constructive approach that removes isomorphisms?

- **Constraint:** any subtree in a tree must be organized so that the main branch with most leaves is on the right

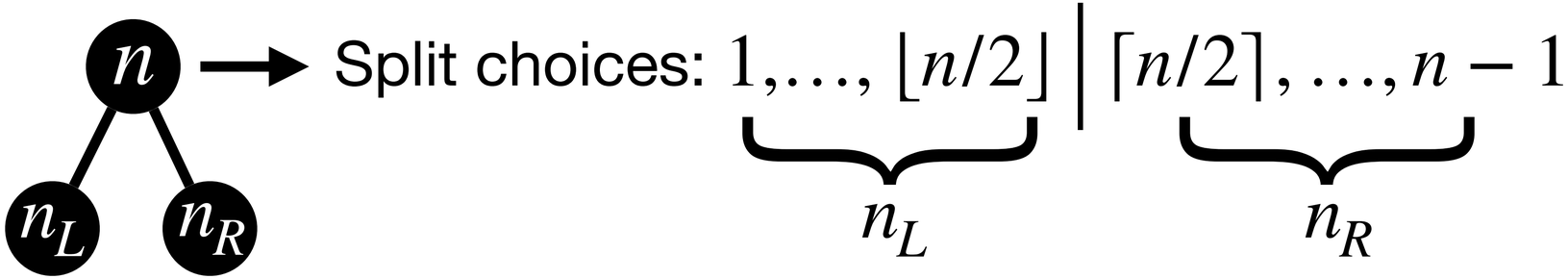


Generating binary tree topologies

First, we need $G_T(n) \rightarrow$ generates a rooted binary tree with n leaves

What about a constructive approach that removes isomorphisms?

- **Constraint:** any subtree in a tree must be organized so that the main branch with most leaves is on the right



- Assign a weight to every possible split $(n_L, n_R) \rightarrow w_i = \max(1, \lfloor n_L/2 \rfloor) \cdot \lfloor n_R/2 \rfloor$
- Then, sample each split with probability $p_i = w_i \,/ \sum_j w_j$

➡ Splits that lead to more topologies are favored

Generating phylogenetic binary trees

Now, we need to assign the length to each edge in the topology

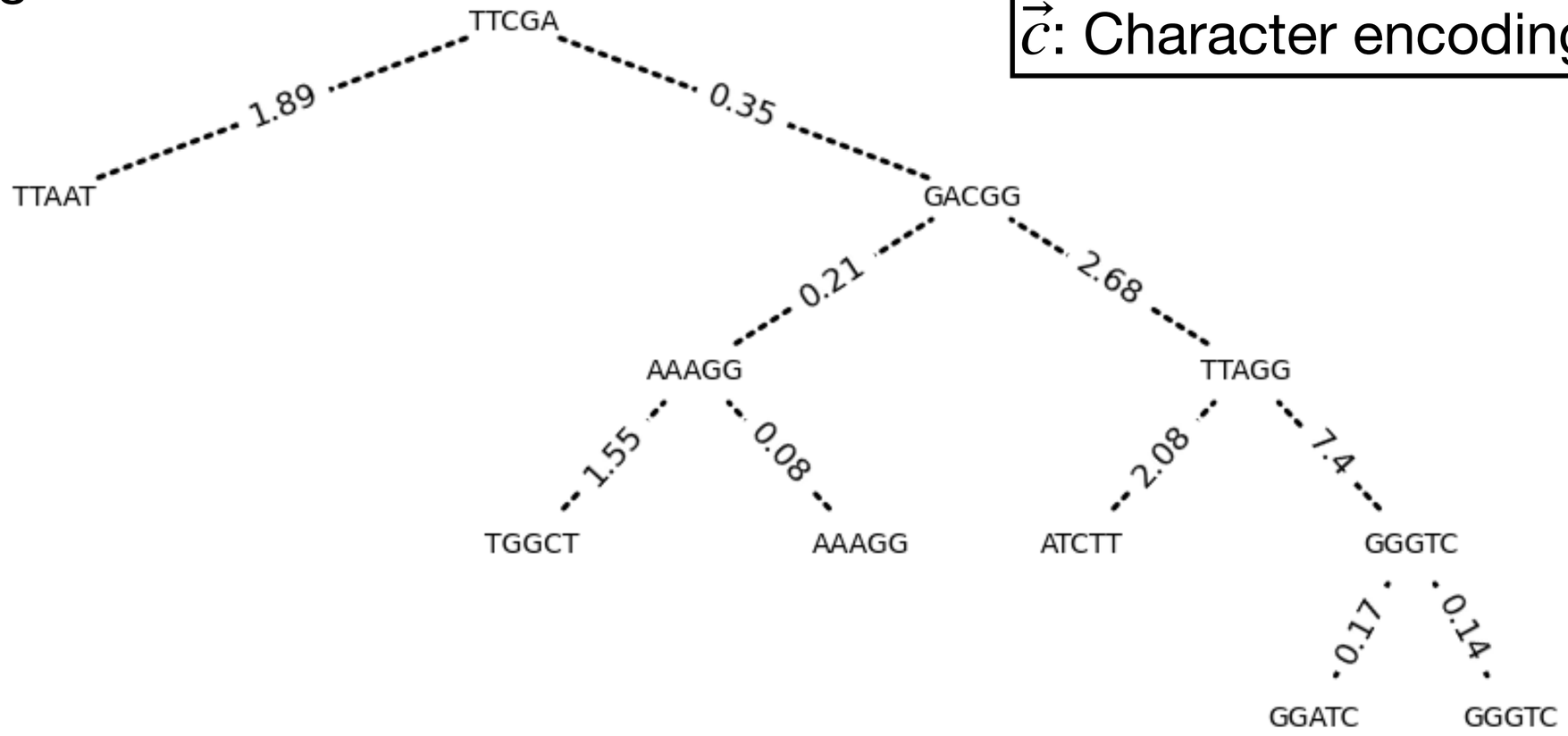
- Same distribution as the prior for inference
- Exponential distribution $\rightarrow P(t) = \lambda \exp(-\lambda t)\ ,\ t \geq 0$
- What is a good choice for the scale parameter λ? $\rightarrow$ More on this later…

Generating phylogenetic binary trees

Finally, we need to generate leaf sequences with length m

- Initialize the root sequence $\rightarrow S_{\rho,i} \sim U(A, G, C, T)\ ,\ i = 1,\ldots,m$
- Draw children sequences $\rightarrow S_{child,i} \sim \vec{p}_i = \vec{c}_{\rho,i} M(t_e)\ ,\ i = 1,\ldots,m$
- Repeat until reaching the leaves

$\vec{c}$: Character encoding

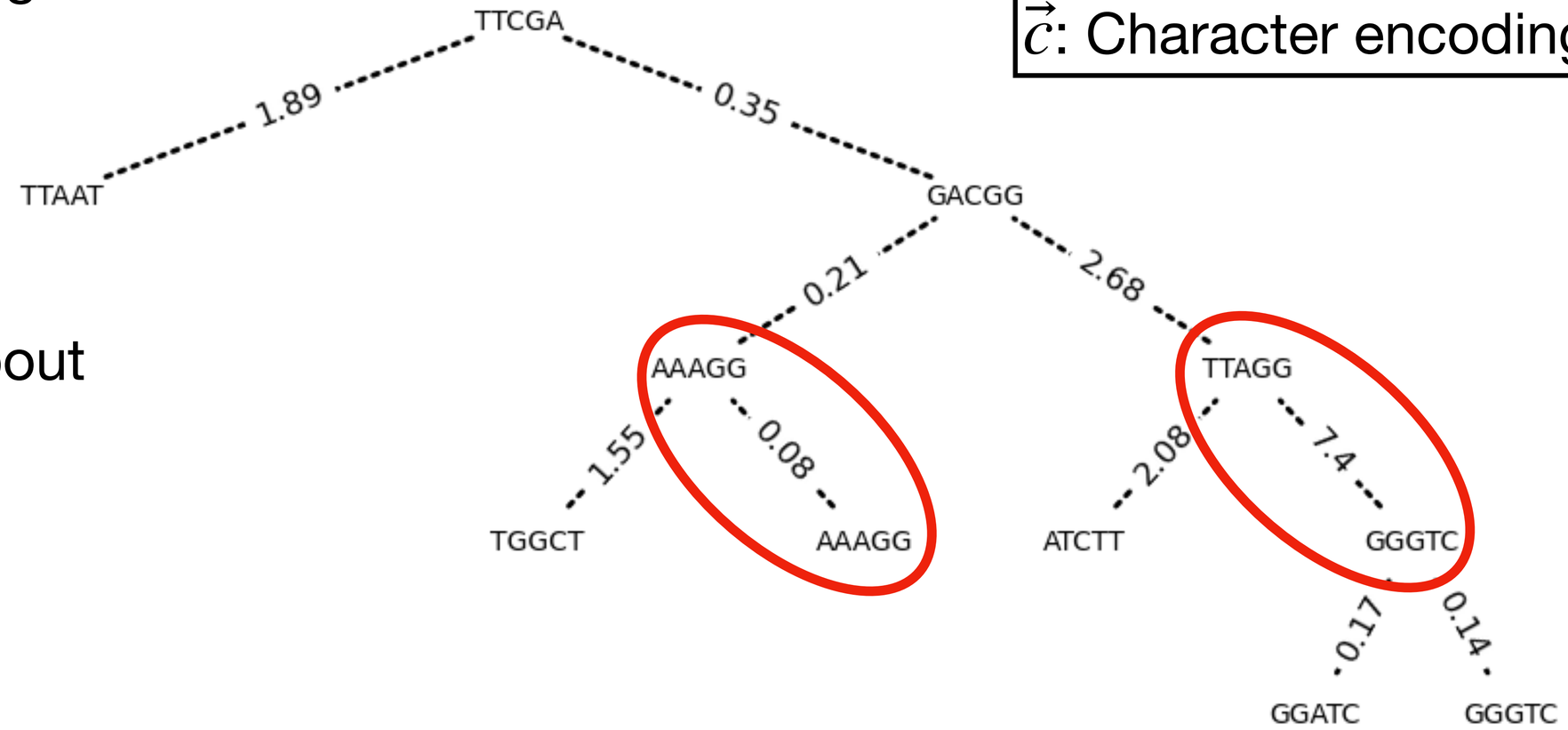


Generating phylogenetic binary trees

Finally, we need to **generate** leaf sequences with length m

- Initialize the root sequence $\rightarrow S_{\rho,i} \sim U(A,G,C,T)\ ,\ i = 1,\ldots,m$
- Draw children sequences $\rightarrow S_{child,i} \sim \vec{p}_i = \vec{c}_{\rho,i} M(t_e)\ ,\ i = 1,\ldots,m$
- Repeat until reaching the leaves

$\vec{c}$: Character encoding

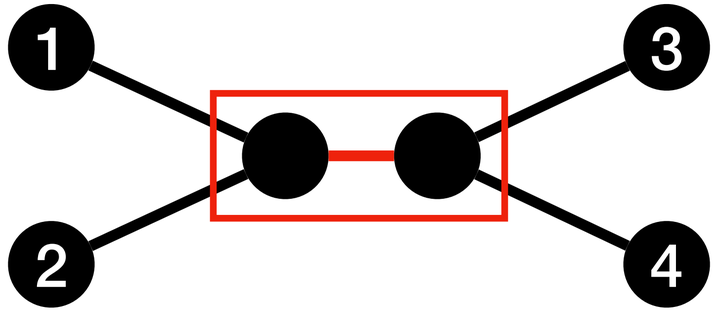


Notice something about the sequences?

Tree moves for MCMC inference

External MCMC step → exploration of tree space

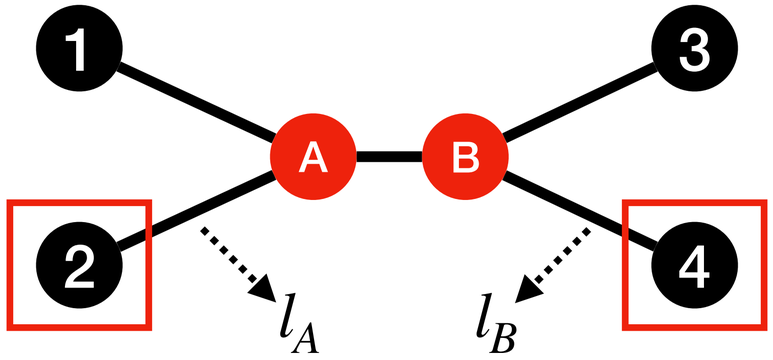
- NNI move:
    1. Choose an internal edge and gather the two nodes



Tree moves for MCMC inference

External MCMC step → exploration of tree space

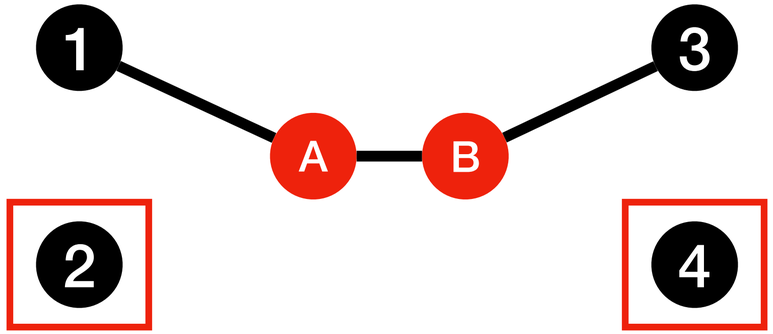
- NNI move:
  1. Choose an internal edge and gather the two nodes
  2. Select one external component for each node uniformly at random, and register edge lengths



Tree moves for MCMC inference

External MCMC step → exploration of tree space

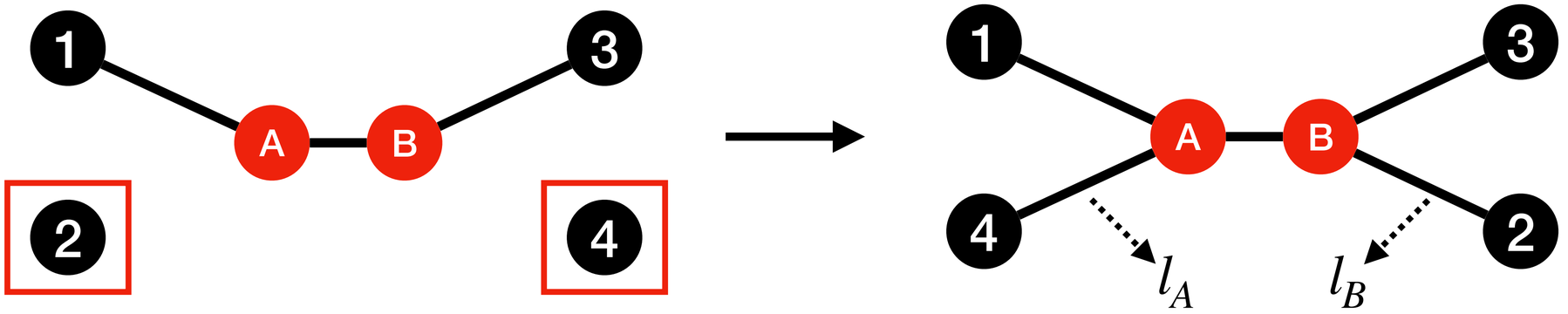
- NNI move:
  1. Choose an internal edge and gather the two nodes
  2. Select one external component for each node uniformly at random, and register edge lengths
  3. Destroy registered edges



Tree moves for MCMC inference

External MCMC step → exploration of tree space

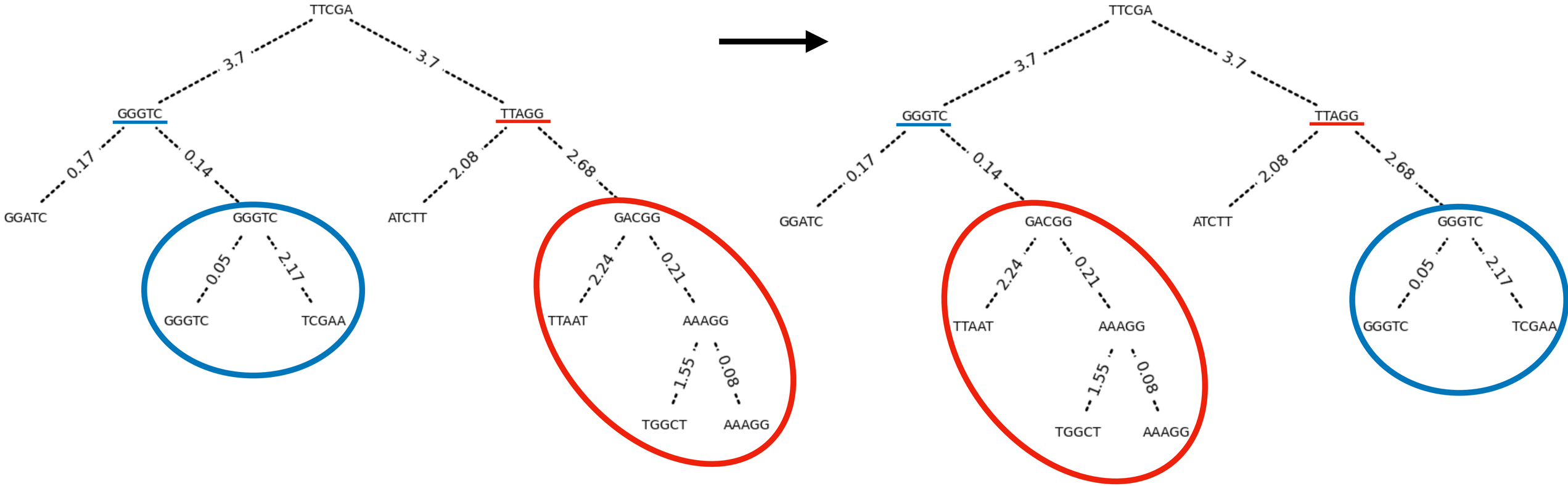
- NNI move:
  1. Choose an internal edge and gather the two nodes
  2. Select one external component for each node uniformly at random, and register edge lengths
  3. Destroy registered edges
  4. Remap components to their new parents, and assign edge length registered for that parent



Tree moves for MCMC inference

External MCMC step → exploration of tree space

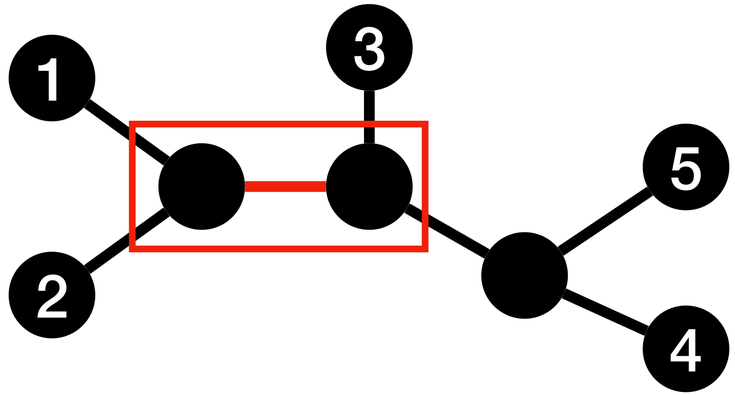
- NNI move:



Tree moves for MCMC inference

External MCMC step → exploration of tree space

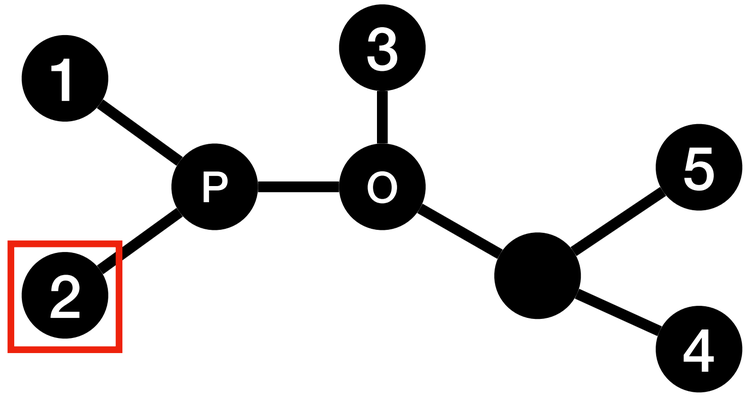
- SPR move:
    1. Choose an internal edge and gather the two nodes



Tree moves for MCMC inference

External MCMC step → exploration of tree space

- SPR move:
  1. Choose an internal edge and gather the two nodes
  2. Define smallest component as “prune” and randomly choose a neighbor

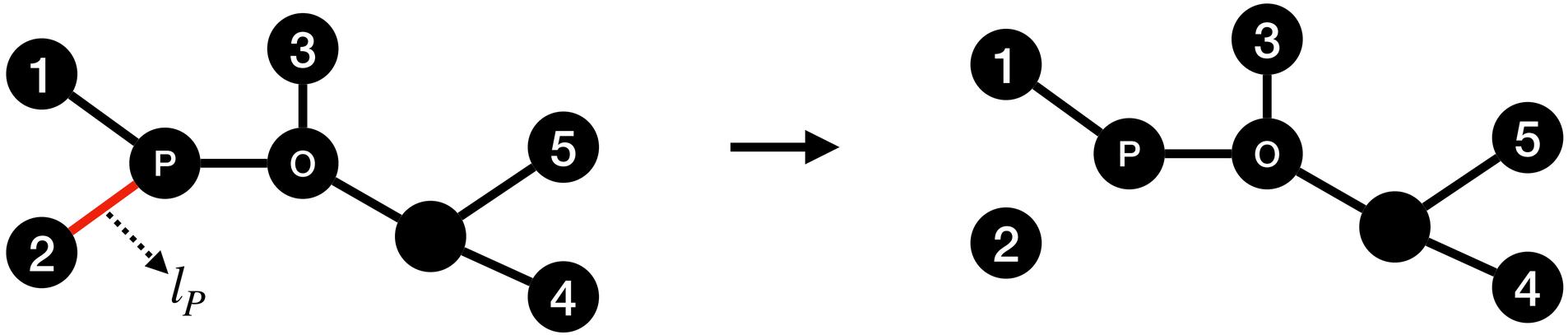


P: prune, O: origin

Tree moves for MCMC inference

External MCMC step → exploration of tree space

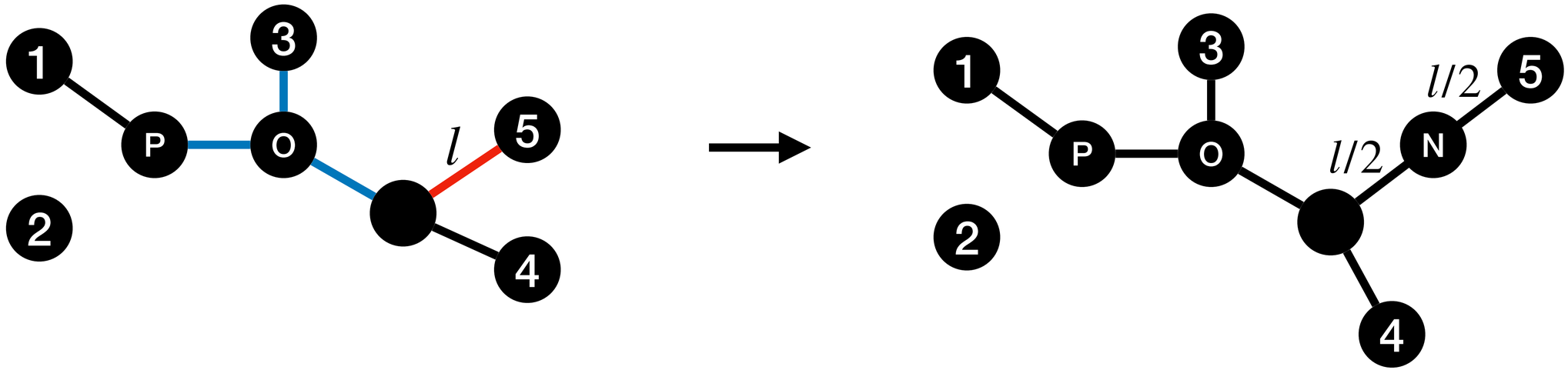
- SPR move:
  1. Choose an internal edge and gather the two nodes
  2. Define smallest component as “prune” and randomly choose a neighbor
  3. Register pruned edge length and remove edge



Tree moves for MCMC inference

External MCMC step → exploration of tree space

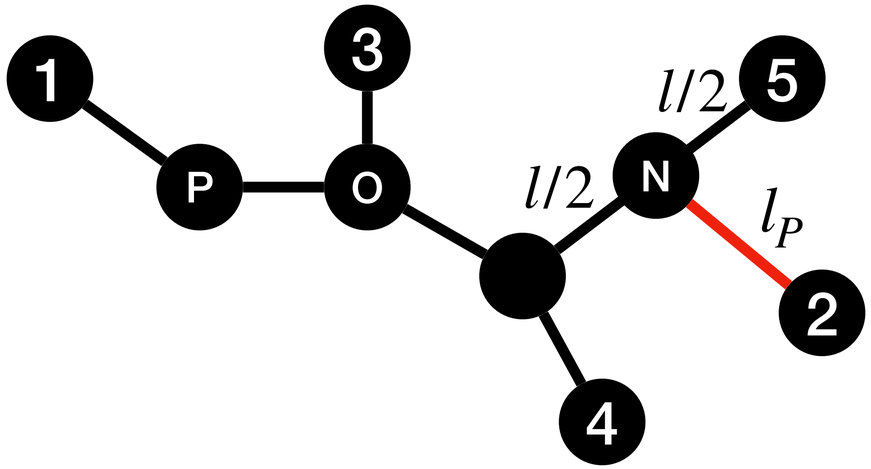
- SPR move:
  1. Choose an internal edge and gather the two nodes
  2. Define smallest component as “prune” and randomly choose a neighbor
  3. Register pruned edge length and remove edge
  4. Select edge for regraft (exclude origin edges), and split in half with a new node



Tree moves for MCMC inference

External MCMC step → exploration of tree space

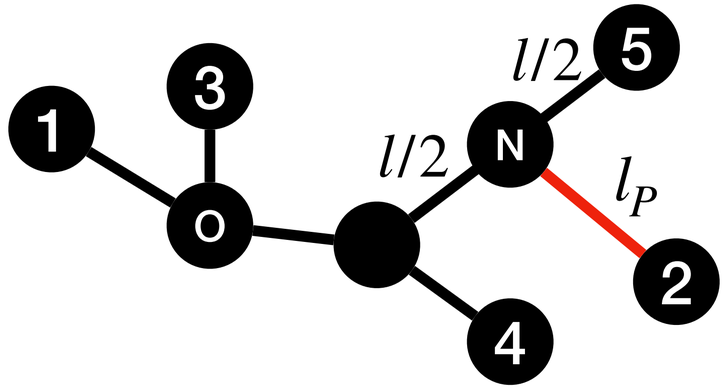
- SPR move:
  1. Choose an internal edge and gather the two nodes
  2. Define smallest component as “prune” and randomly choose a neighbor
  3. Register pruned edge length and remove edge
  4. Select edge for regraft (exclude origin edges), and split in half with a new node
  5. Regraft pruned subtree and assigned the registered length to the edge



Tree moves for MCMC inference

External MCMC step → exploration of tree space

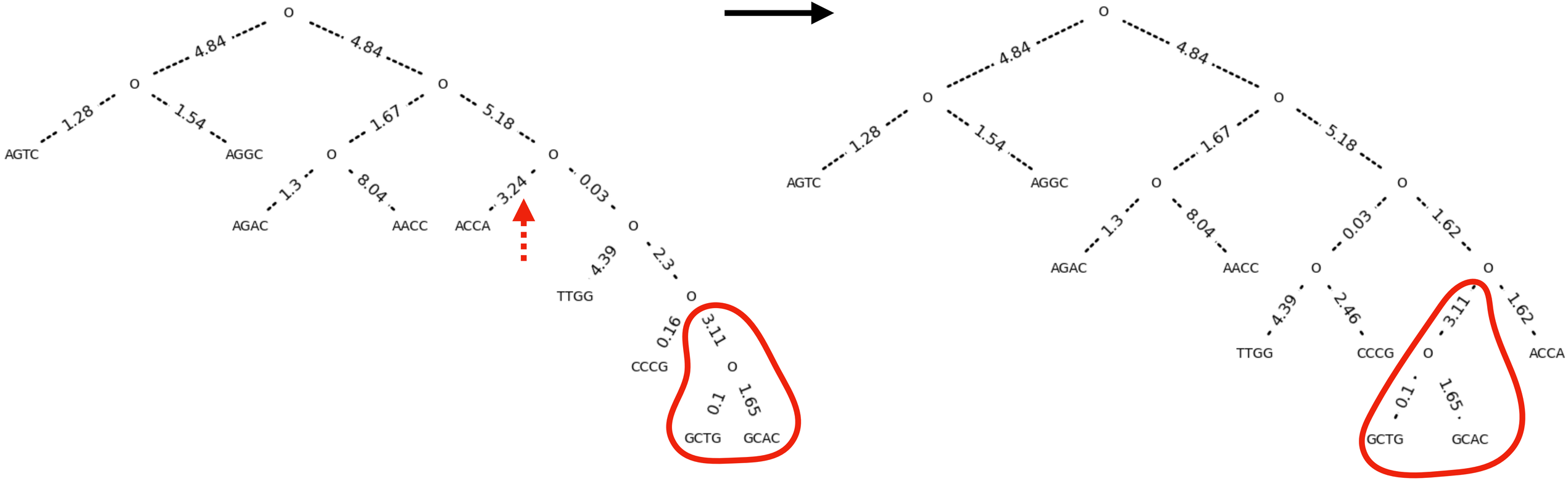
- SPR move:
  1. Choose an internal edge and gather the two nodes
  2. Define smallest component as “prune” and randomly choose a neighbor
  3. Register pruned edge length and remove edge
  4. Select edge for regraft (exclude origin edges), and split in half with a new node
  5. Regraft pruned subtree and assigned the registered length to the edge
  6. Simplify → remove redundant nodes



Tree moves for MCMC inference

External MCMC step → exploration of tree space

- SPR move:



Edge length exploration for MCMC inference

Internal MCMC step → exploration of internal degrees of freedom (edge lengths)

- Initializing edges with *reasonable* values
- Defining the edge length prior, $P(t)$
- Defining the edge length proposal distribution, $P(t' \mid t)$

But first, we need to get a better understanding of the effect of the edge length…

Edge length exploration for MCMC inference

Internal MCMC step → exploration of internal degrees of freedom (edge lengths)

- Initializing edges with *reasonable* values
- Defining the edge length prior, $P(t)$
- Defining the edge length proposal distribution, $P(t' \mid t)$

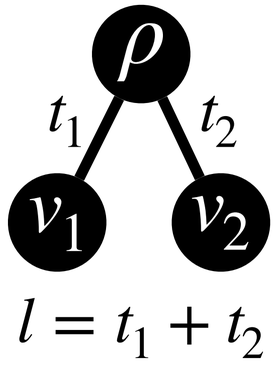
But first, we need to get a better understanding of the effect of the edge length…

Note → JC model:

- α and t only appear as αt in $a(t) = \frac{3}{4}\left(1 - \exp\left(-\frac{4}{3}\alpha t\right)\right)$
- Since we cannot infer α independently, we set $\alpha = 1$

Effect of the edge length on evolution

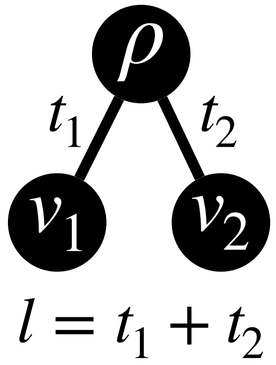
Consider a simple scenario → tree with a root and two leaves



- v_1, v_2 encode 1-character sequences → example $S_1 = A, S_2 = G$

Effect of the edge length on evolution

Consider a simple scenario → tree with a root and two leaves



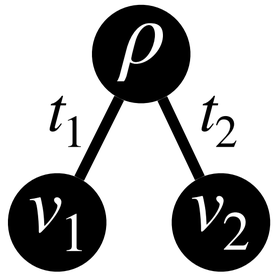
- v_1, v_2 encode 1-character sequences → example $S_1 = A, S_2 = G$

- For $S_1 \neq S_2$, the likelihood can be shown to be:

$$L = \frac{1}{16}\left[1 - \exp\left(-\frac{4}{3}l\right)\right]$$

Effect of the edge length on evolution

Consider a simple scenario → tree with a root and two leaves



- v_1, v_2 encode 1-character sequences → example $S_1 = A, S_2 = G$

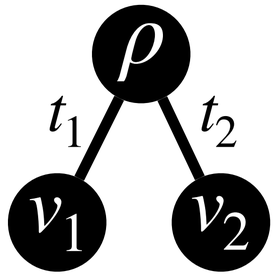
- For $S_1 \neq S_2$, the likelihood can be shown to be: $$L = \frac{1}{16}\left[1 - \exp\left(-\frac{4}{3}l\right)\right]$$

  - $l \to 0 \Rightarrow L = 0$, because it is impossible for the sequences to be different
  - $l \to \infty \Rightarrow L \to 1/16$, any possible outcome is equally likely
  - $L = 1/16$ is the maximum likelihood for $S_1 \neq S_2$

➡ $l \to \infty$ is a “lazy” maximum, because it does not “commit” to a relationship between S_1 and S_2.

Effect of the edge length on evolution

Consider a simple scenario → tree with a root and two leaves



$l = t_1 + t_2$

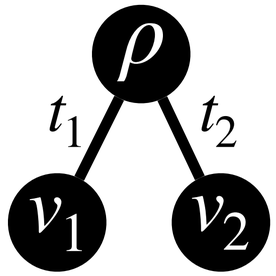
- v_1, v_2 encode 1-character sequences → example $S_1 = A, S_2 = G$

- For $S_1 = S_2$, the likelihood can be shown to be:

$$L = \frac{1}{16}\left[1 + 3\exp\left(-\frac{4}{3}l\right)\right]$$

Effect of the edge length on evolution

Consider a simple scenario → tree with a root and two leaves

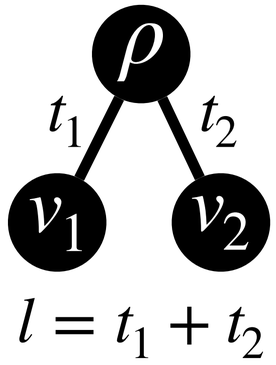


- v_1, v_2 encode 1-character sequences → example $S_1 = A, S_2 = G$

- For $S_1 = S_2$, the likelihood can be shown to be: $$L = \frac{1}{16}\left[1 + 3\exp\left(-\frac{4}{3}l\right)\right]$$

  - $l \to 0 \Rightarrow L = 1/4$, the chance that $S_\rho = S_1 = S_2$
  - $l \to \infty \Rightarrow L \to 1/16$, any possible outcome is equally likely
  - $L = 1/4$ is the maximum likelihood for $S_1 = S_2$

Effect of the edge length on evolution

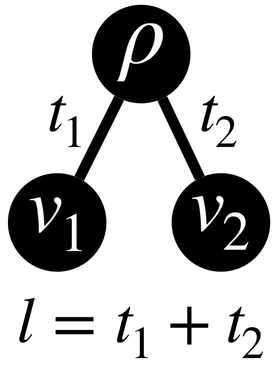


Consider a simple scenario → tree with a root and two leaves

- Let us consider S_1, S_2 with n characters. If S_1, S_2 have n^+ matching characters and n^- mismatched characters, then the likelihood is:

$$L = \left(\frac{1}{16}\right)^n \left(1 + 3e^{-4l/3}\right)^{n^+} \left(1 - e^{-4l/3}\right)^{n^-}$$

Effect of the edge length on evolution



Consider a simple scenario → tree with a root and two leaves

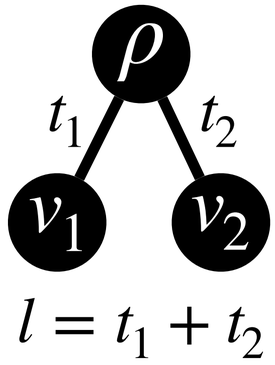
- Let us consider S_1, S_2 with n characters. If S_1, S_2 have n^+ matching characters and n^- mismatched characters, then the likelihood is:

$$L = \left(\frac{1}{16}\right)^n \left(1 + 3e^{-4l/3}\right)^{n^+} \left(1 - e^{-4l/3}\right)^{n^-}$$

  - The maximum likelihood is given by: $l^* = -\frac{3}{4} \ln \left[\frac{1}{3n}(n^+ - 3n^-)\right]$
  - No solution if $n^+ \leq 3n^-$

- We can already use this notion to initialize edge lengths to **reasonable** values.

Effect of the edge length on evolution

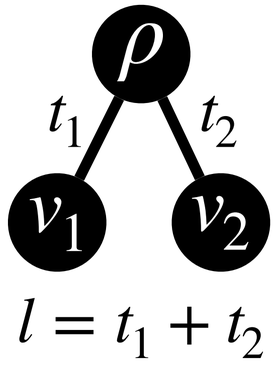


Let us generate *useful* synthetic data from our new understanding

- JC model → What is the probability that we generate $S_1 = S_2$?
  Assume we have $t_1 = t_2 = l/2$...

$$P(S_1 = S_2) = 1 - 2a(l/2) + \frac{4}{3}a^2(l/2)$$

Effect of the edge length on evolution



Let us generate *useful* synthetic data from our new understanding

- JC model → What is the probability that we generate $S_1 = S_2$?
  Assume we have $t_1 = t_2 = l/2$...

$$P(S_1 = S_2) = 1 - 2a(l/2) + \frac{4}{3}a^2(l/2)$$

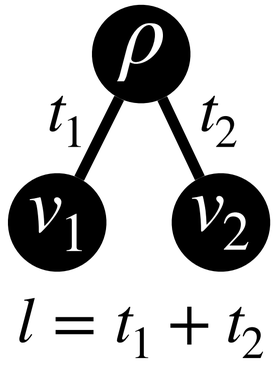
- Since the process is independent for each site, then n^+ follows a *Binomial* distribution with mean:

$$\mathbb{E}[n^+] = nP(S_1 = S_2)$$

- Our condition for a non-“lazy” maximum-likelihood was: $n^+ > 3n^- \Leftrightarrow n^+ > 3n/4$

  ➡ It translates to $P(S_1 = S_2) > 3/4$

Effect of the edge length on evolution



Let us generate *useful* synthetic data from our new understanding

- JC model → What is the probability that we generate $S_1 = S_2$?
  Assume we have $t_1 = t_2 = l/2$...

$$P(S_1 = S_2) = 1 - 2a(l/2) + \frac{4}{3}a^2(l/2)$$

- Since the process is independent for each site, then n^+ follows a *Binomial* distribution with mean:

$$\mathbb{E}[n^+] = nP(S_1 = S_2)$$

- Our condition for a non-“lazy” maximum-likelihood was: $n^+ > 3n^- \Leftrightarrow n^+ > 3n/4$

  ➡ It translates to $P(S_1 = S_2) > 3/4$

- **Upper boundary** to generate synthetic data that can be reconstructed with inference:

$$l^* < -\frac{3}{4}\ln\left(\frac{2}{3}\right)$$

Distribution of edge lengths

From the upper bound we found, we may propose the edge length distribution

$$P(t) = \lambda e^{-\lambda t} \quad ; \quad \lambda = \frac{1}{t^* + \delta_t} \quad ; \quad t^* = \frac{l^*}{2} = -\frac{3}{8} \ln\left(\frac{2}{3}\right) \approx 0.15$$

Distribution of edge lengths

From the upper bound we found, we may propose the edge length distribution

$$P(t) = \lambda e^{-\lambda t} \quad ; \quad \lambda = \frac{1}{t^* + \delta_t} \quad ; \quad t^* = \frac{l^*}{2} = -\frac{3}{8} \ln\left(\frac{2}{3}\right) \approx 0.15$$

- Used for data generation and as prior for inference
- δ_t determines how closely related the daughter and parent sequences are:
  - $\delta_t > 0 \rightarrow n^+/n^-$ decreases and sequences are less correlated
  - $\delta_t < 0 \rightarrow n^+/n^-$ increases and sequences are more correlated

Proposal distribution for edge lengths

We want the following qualities from our proposal:

- Symmetric $\rightarrow P(t' \mid t) = P(t \mid t')$
- Avoids proposing negative values $\rightarrow P(t' < 0 \mid t) = 0, \forall t$

Let us propose $t' = t \cdot e^{\gamma}$, with $\gamma \sim U(-\gamma_0, \gamma_0)$

➡ Then, $P(t' \mid t) = P(\gamma = \ln(t'/t))$ and fulfills our constraints

Complete MCMC inference scheme

1. Initialize tree topology and edge lengths from $P(t)$.
2. Propose N_{seq} sequence assignments for the tree and keep the one with maximum likelihood.
3. Perform N_{burn} MC steps for burn-in.
4. Perform $N_{samples}$ MC steps for sampling.

Complete MCMC inference scheme

1. Initialize tree topology and edge lengths from $P(t)$.
2. Propose N_{seq} sequence assignments for the tree and keep the one with maximum likelihood.
3. Perform N_{burn} MC steps for burn-in.
4. Perform $N_{samples}$ MC steps for sampling.

Each MC step consists of:

a. **External** MC step: attempt $N_{int,e}$ NNI moves and N_{SPR} SPR moves.

b. **Internal** MC step: attempt N_e edge changes, picked uniformly at random.

c. **Register** sample for the specific topology.

Complete MCMC inference scheme

Let us define the following log-ratios:

$$\Delta_P = \ln[P(S\,|\,T', \{t'_e\})] - \ln[P(S\,|\,T, \{t_e\})] \qquad \Delta_t = \ln[\exp(t')] - \ln[\exp(t)] = t' - t$$

Then, the acceptance probabilities are given by:

- External MC steps ($\{t'_e\} = \{t_e\}$) $\rightarrow$ $A_{ext} = \exp(\Delta_P)$
- Internal MC steps ($T' = T$) $\rightarrow$ $A_{int} = \exp(\Delta_P - \lambda\Delta_t)$

Complete MCMC inference scheme

Let us define the following log-ratios:

$$\Delta_P = \ln[P(S\,|\,T', \{t'_e\})] - \ln[P(S\,|\,T, \{t_e\})] \qquad \Delta_t = \ln[\exp(t')] - \ln[\exp(t)] = t' - t$$

Then, the acceptance probabilities are given by:

- External MC steps ($\{t'_e\} = \{t_e\}$) $\rightarrow$ $A_{ext} = \exp(\Delta_P)$
- Internal MC steps ($T' = T$) $\rightarrow$ $A_{int} = \exp(\Delta_P - \lambda\Delta_t)$

In this context, λ can be interpreted as modulating the reward of shorter edges (“bold” claims for the data) and the cost of longer edges (“lazy” claims).

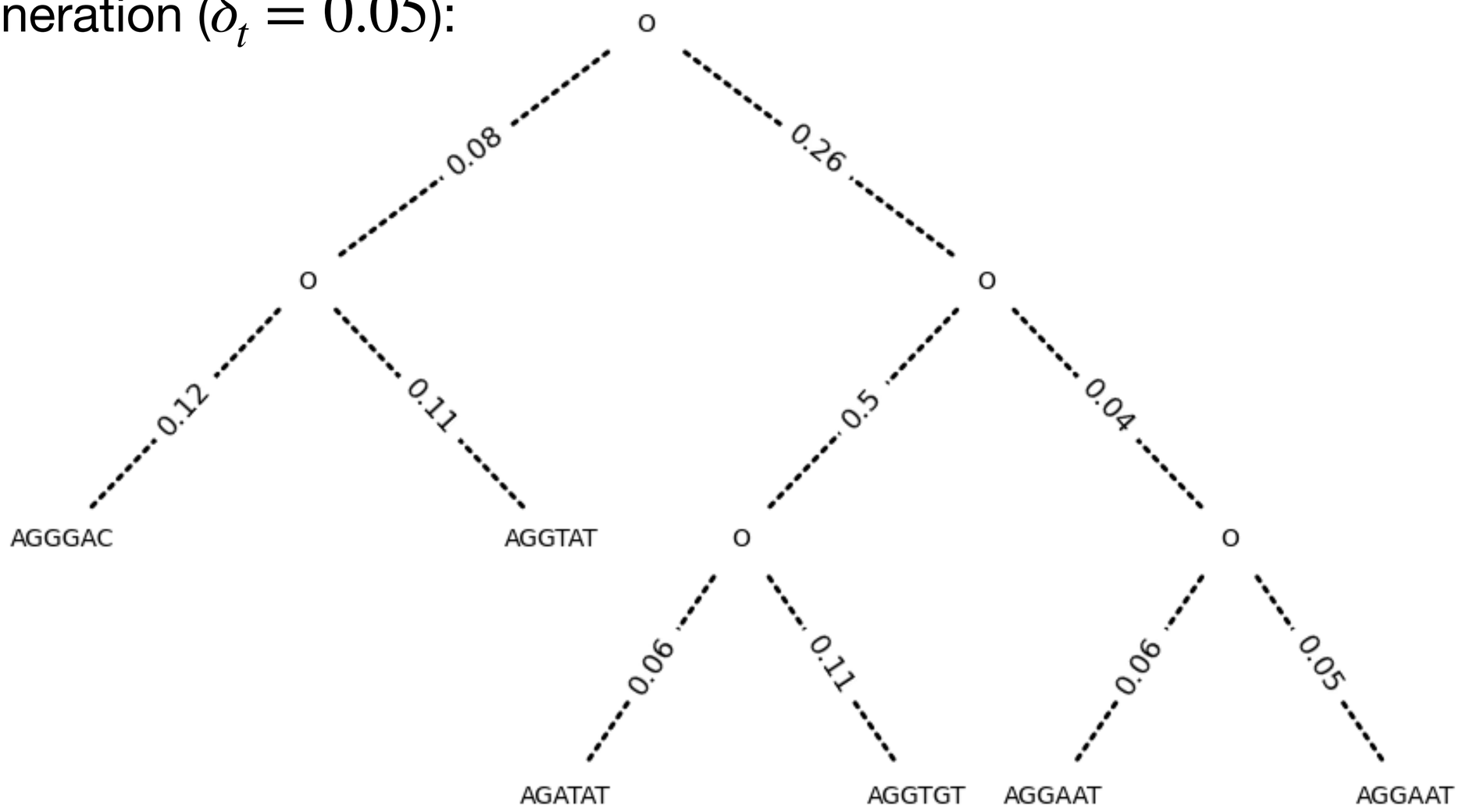
RESULTS

Initial testing

Small test case $\rightarrow n = 6$ leaves, with $m = 6$ sites per sequence

- Synthetic data generation ($\delta_t = 0.05$):

  ➡ $\langle t \rangle \approx 0.2$



Initial testing

Small test case $\rightarrow n = 6$ leaves, with $m = 6$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 2.5, N_{SPR} = 2$

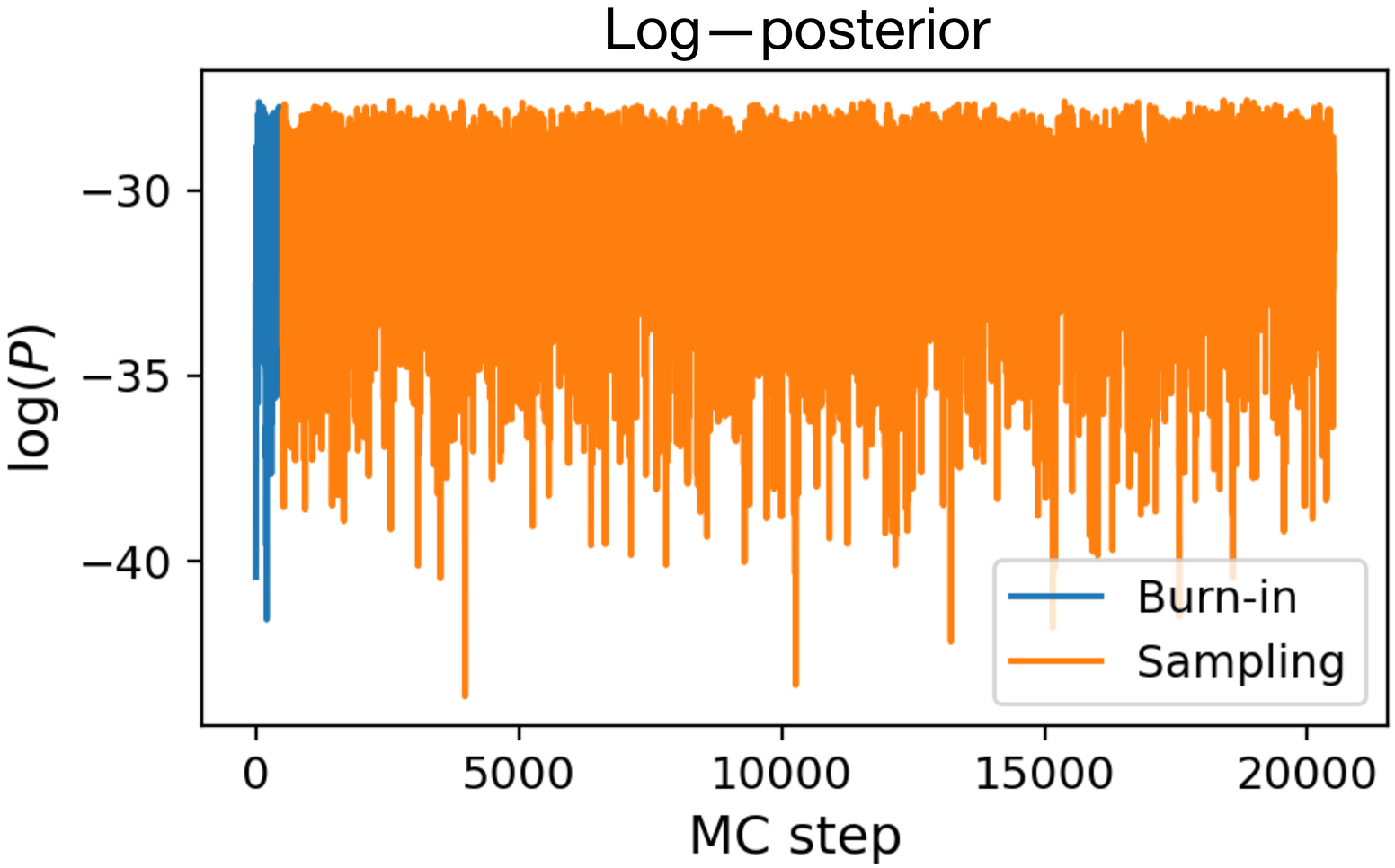
Execution time ≈ 100 s

Initial testing

Small test case $\rightarrow n = 6$ leaves, with $m = 6$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 2.5, N_{SPR} = 2$

Execution time ≈ 100 s

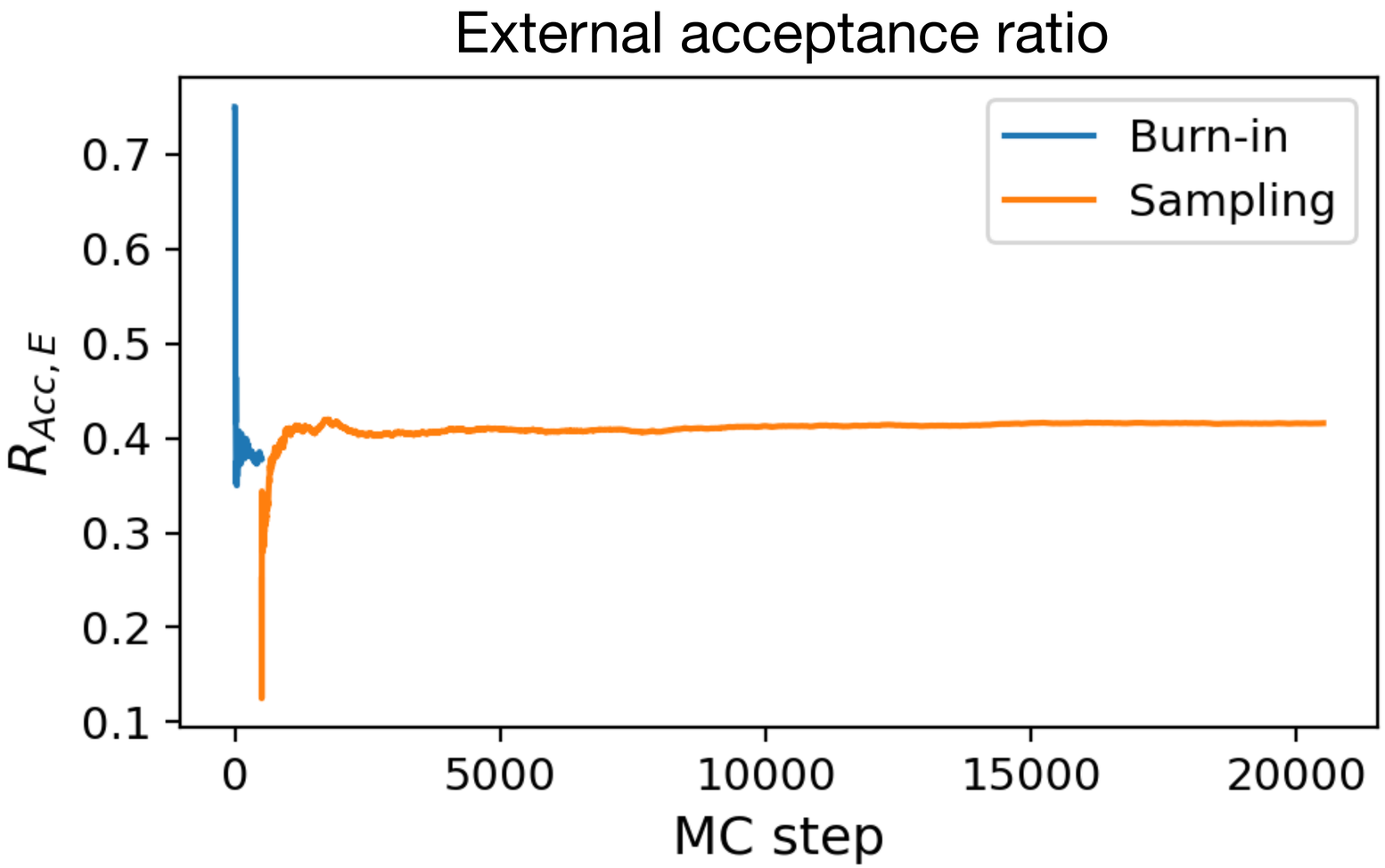


Initial testing

Small test case $\rightarrow n = 6$ leaves, with $m = 6$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 2.5, N_{SPR} = 2$

Execution time ≈ 100 s

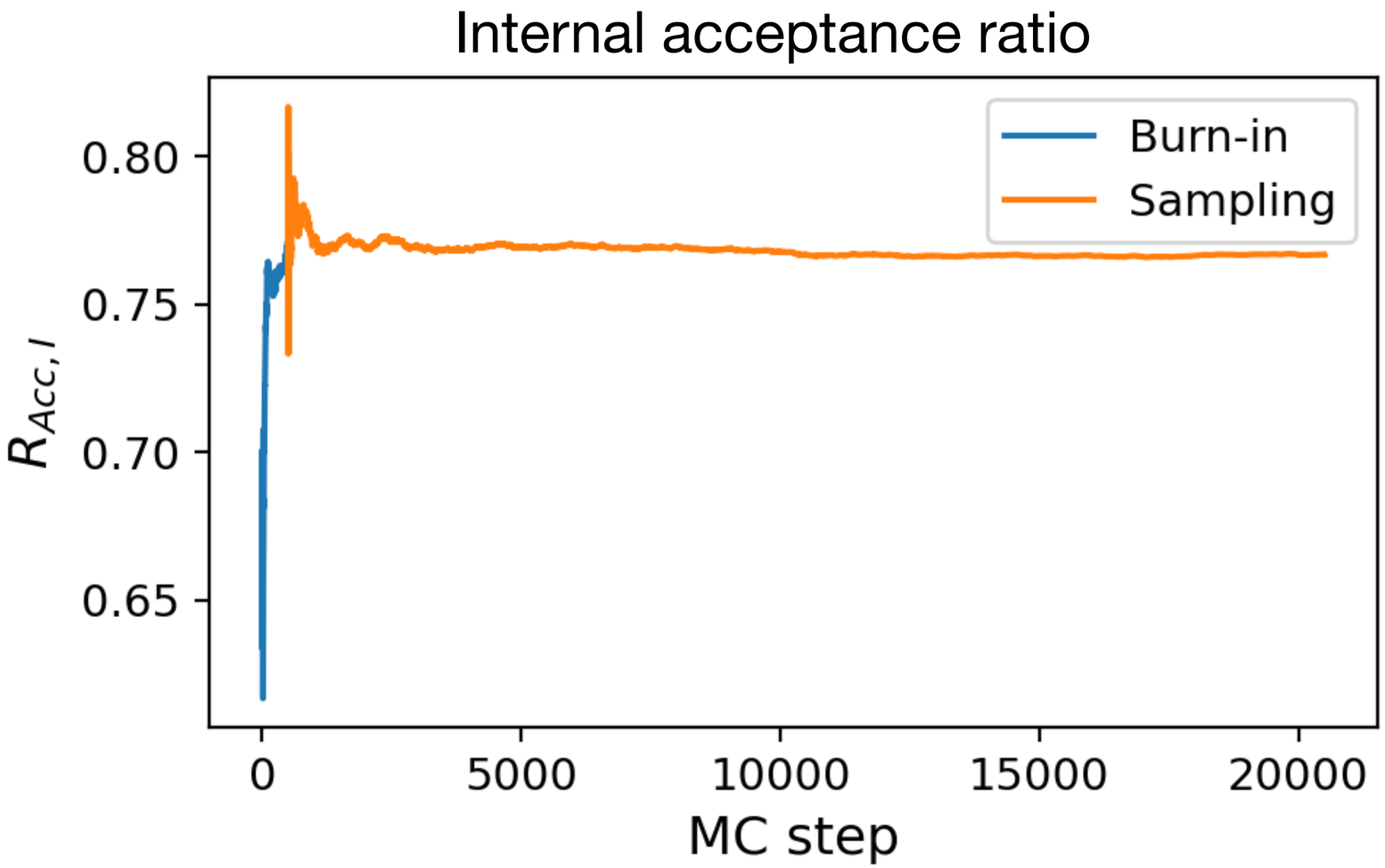


Initial testing

Small test case $\rightarrow n = 6$ leaves, with $m = 6$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 2.5, N_{SPR} = 2$

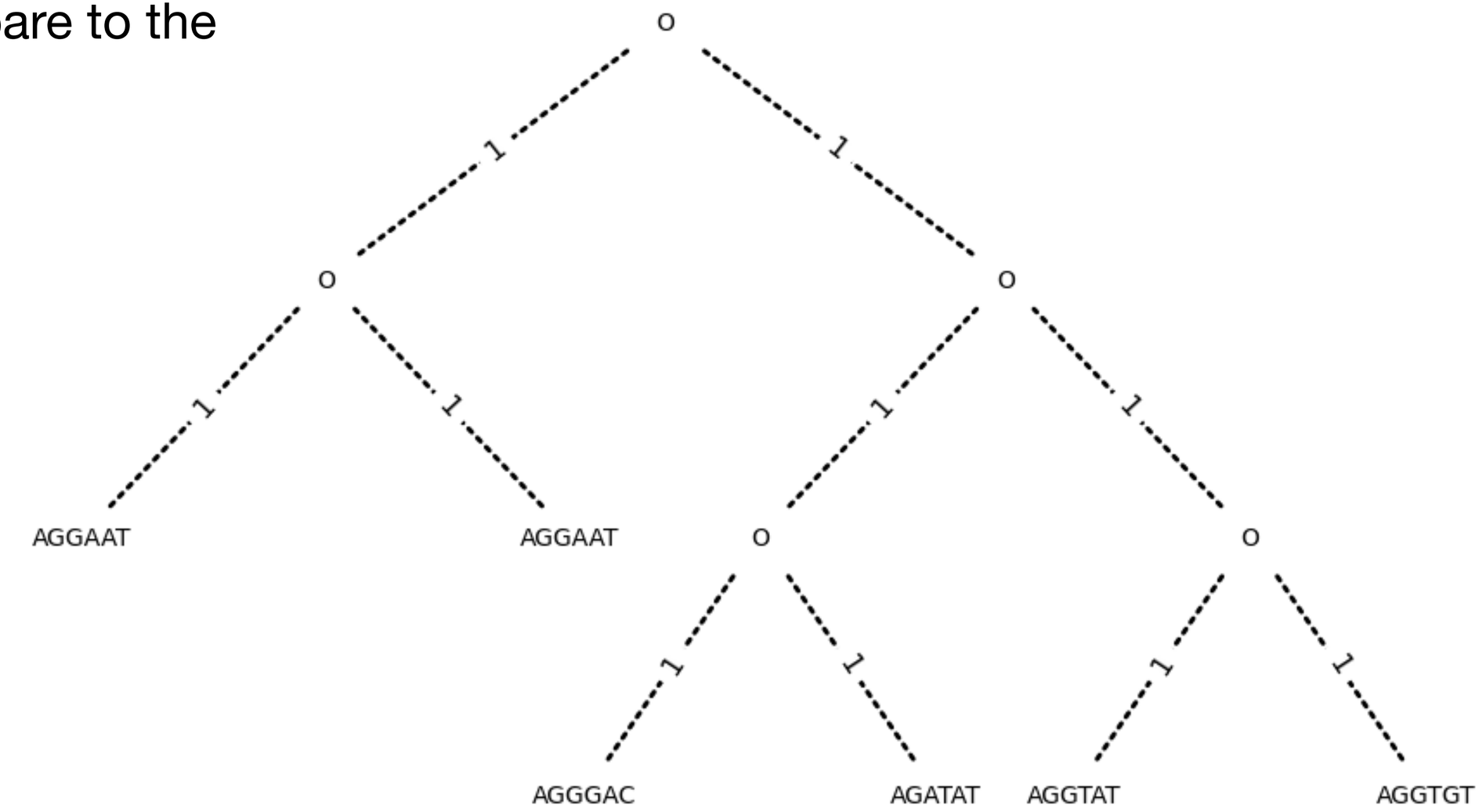
Execution time ≈ 100 s



Initial testing

Maximum a posteriori (MAP) tree → Sampled topology with highest posterior

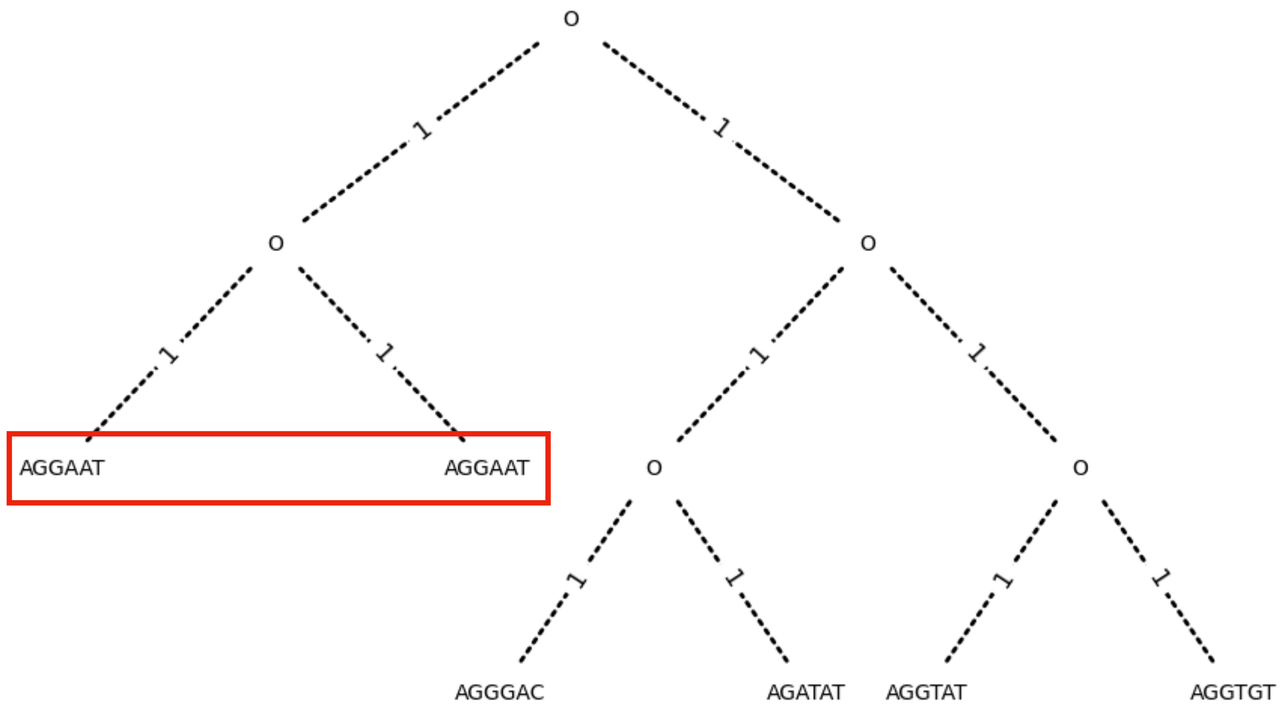
- How does it compare to the original tree?



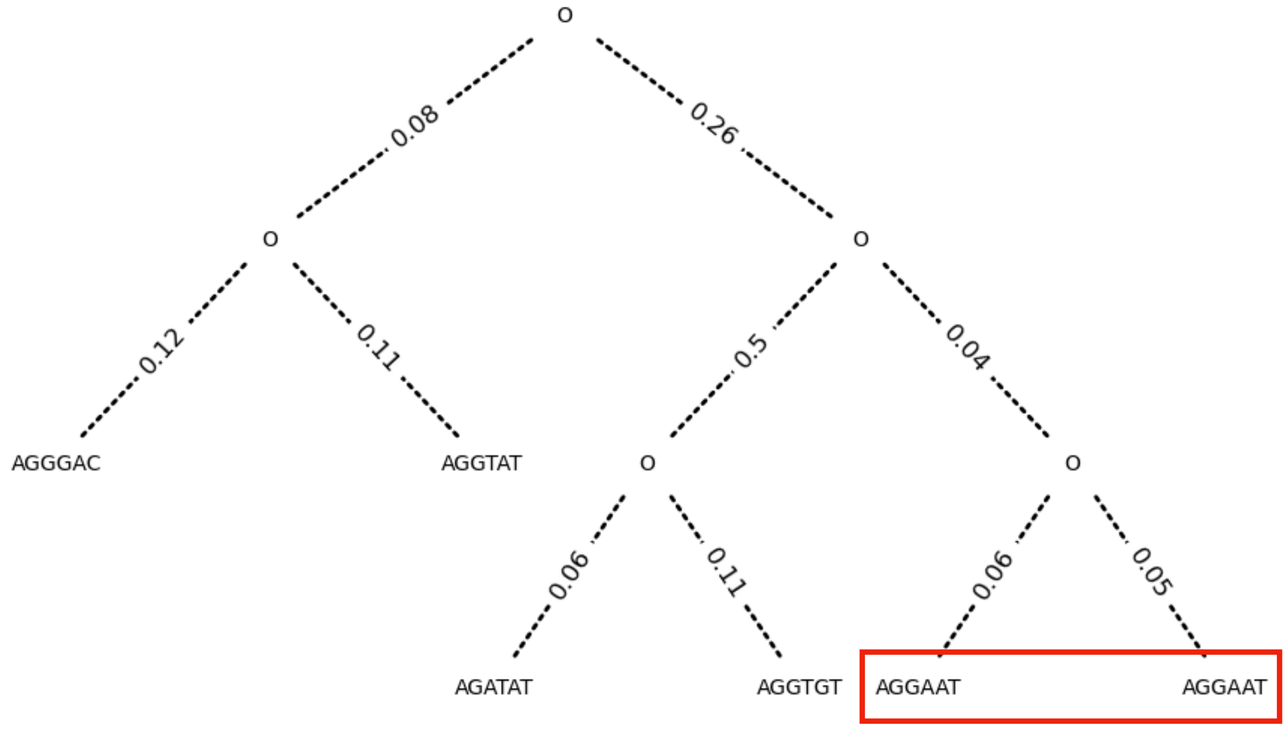
Initial testing

Maximum a posteriori (MAP) tree → Sampled topology with highest posterior

MAP



Original



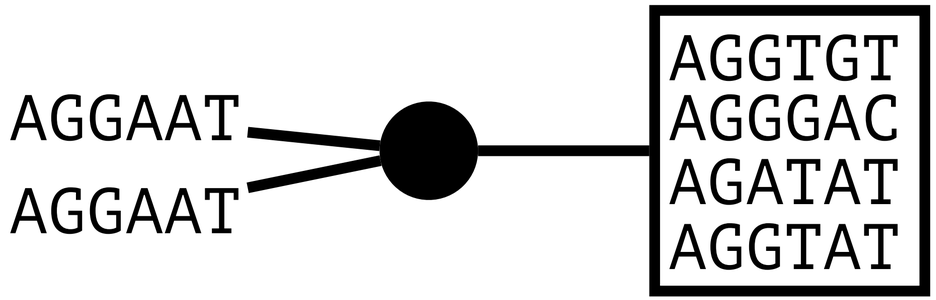
Ranking by $\log(P)$: 259 out of 267

Initial testing

Splits $\rightarrow$ any partition of X into two non—empty disjoint subsets

- Majority—rule consensus: accept the splits that appear in $> 50\,\%$ of the trees

```
((AGGAAT,AGGAAT),(AGGTGT,AGGGAC,AGATAT,AGGTAT))
```



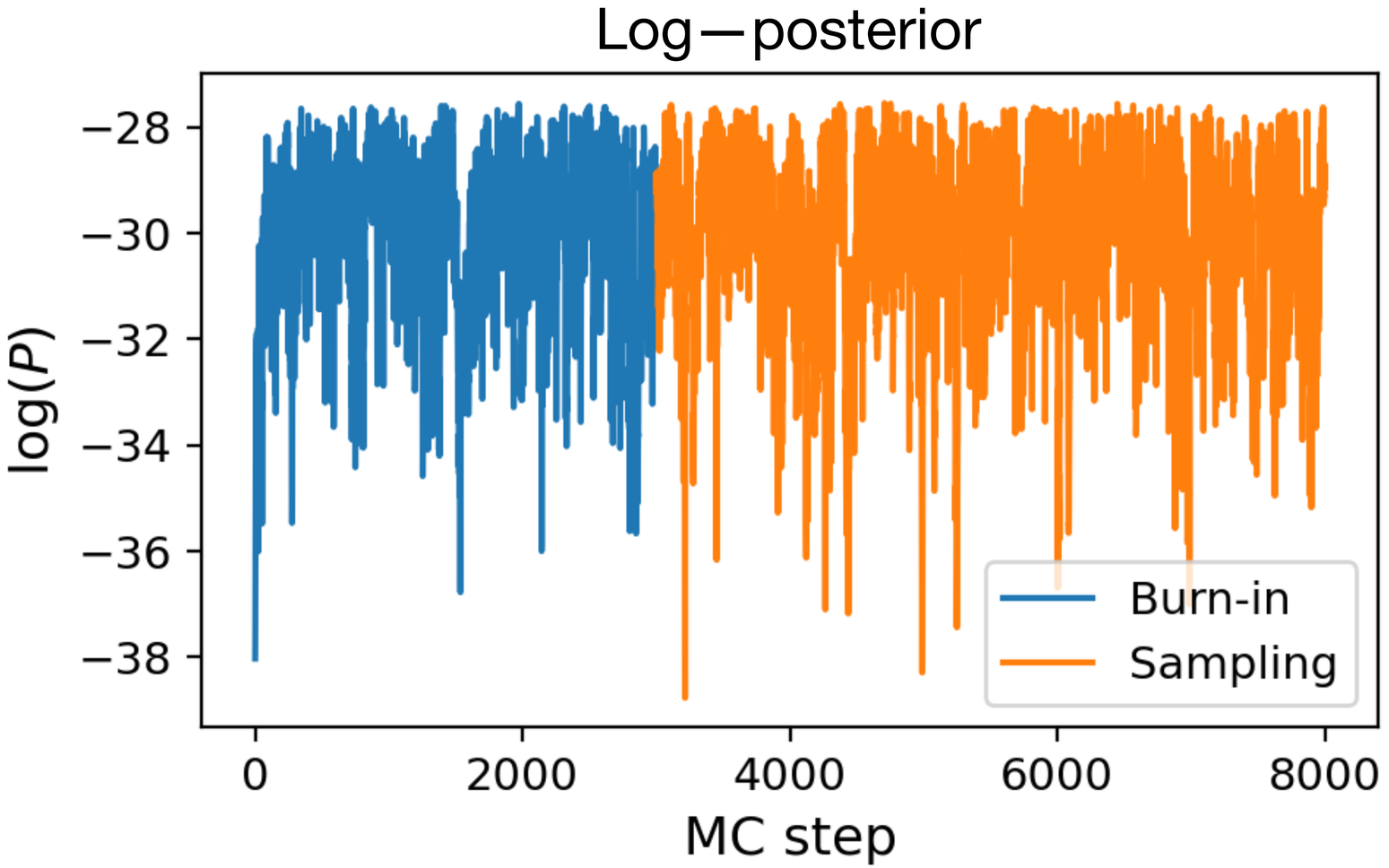
- Original splits:

```
((AGGAAT,AGGAAT),(AGGTGT,AGGGAC,AGATAT,AGGTAT))
((AGGGAC,AGGTAT),(AGGAAT,AGATAT,AGGTGT,AGGAAT))
((AGATAT,AGGTGT),(AGGAAT,AGGGAC,AGGTAT,AGGAAT))
```

Initial testing

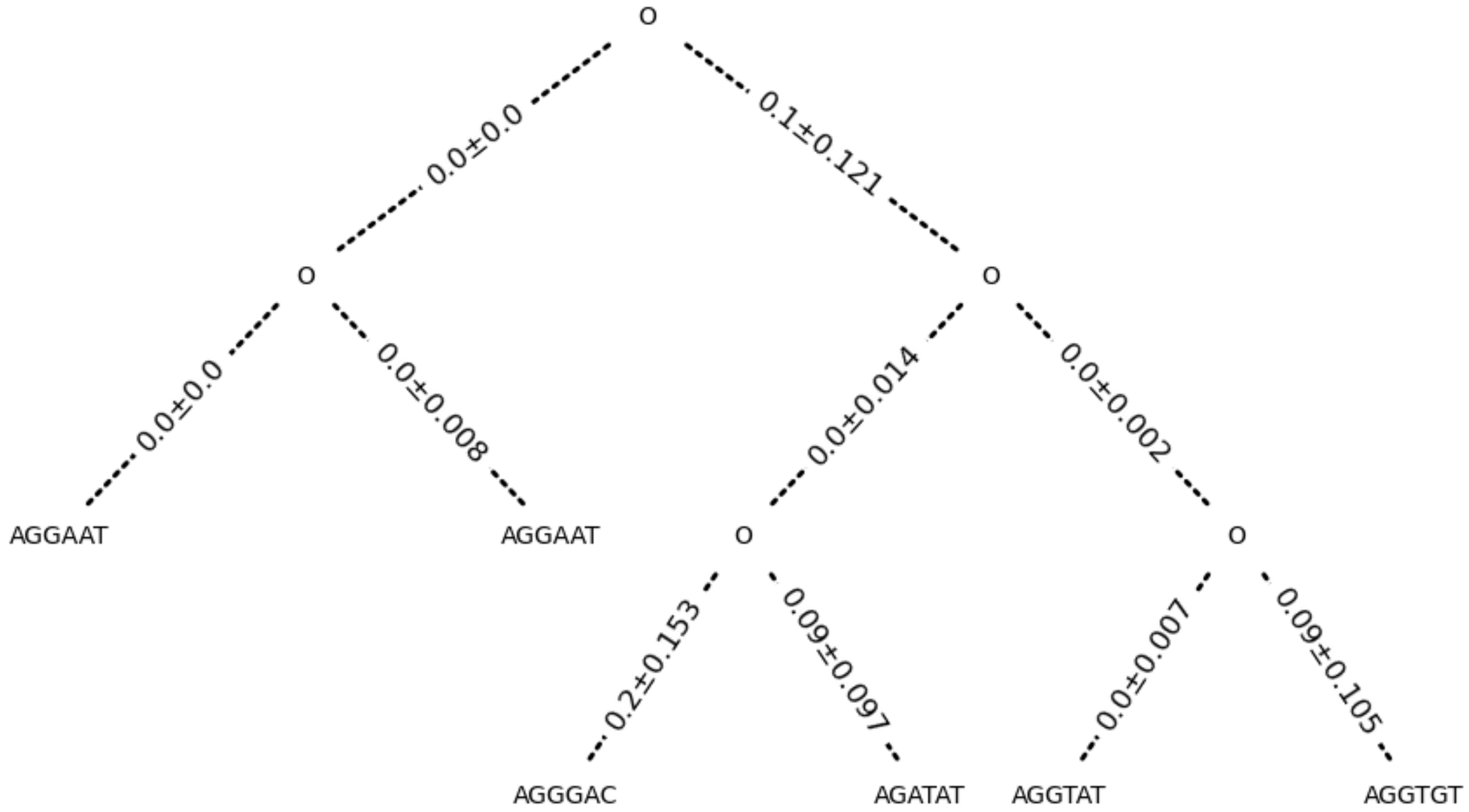
Inference of edge lengths for the MAP tree → Sample with *internal* MC steps

- MCMC inference:
  - 3×10^3 burn-in steps
  - 5×10^3 sampling steps
  - $\gamma_0 = 1$
- Use 1 every 10 samples to reduce correlation.



Initial testing

Inference of edge lengths for the MAP tree → Sample with *internal* MC steps

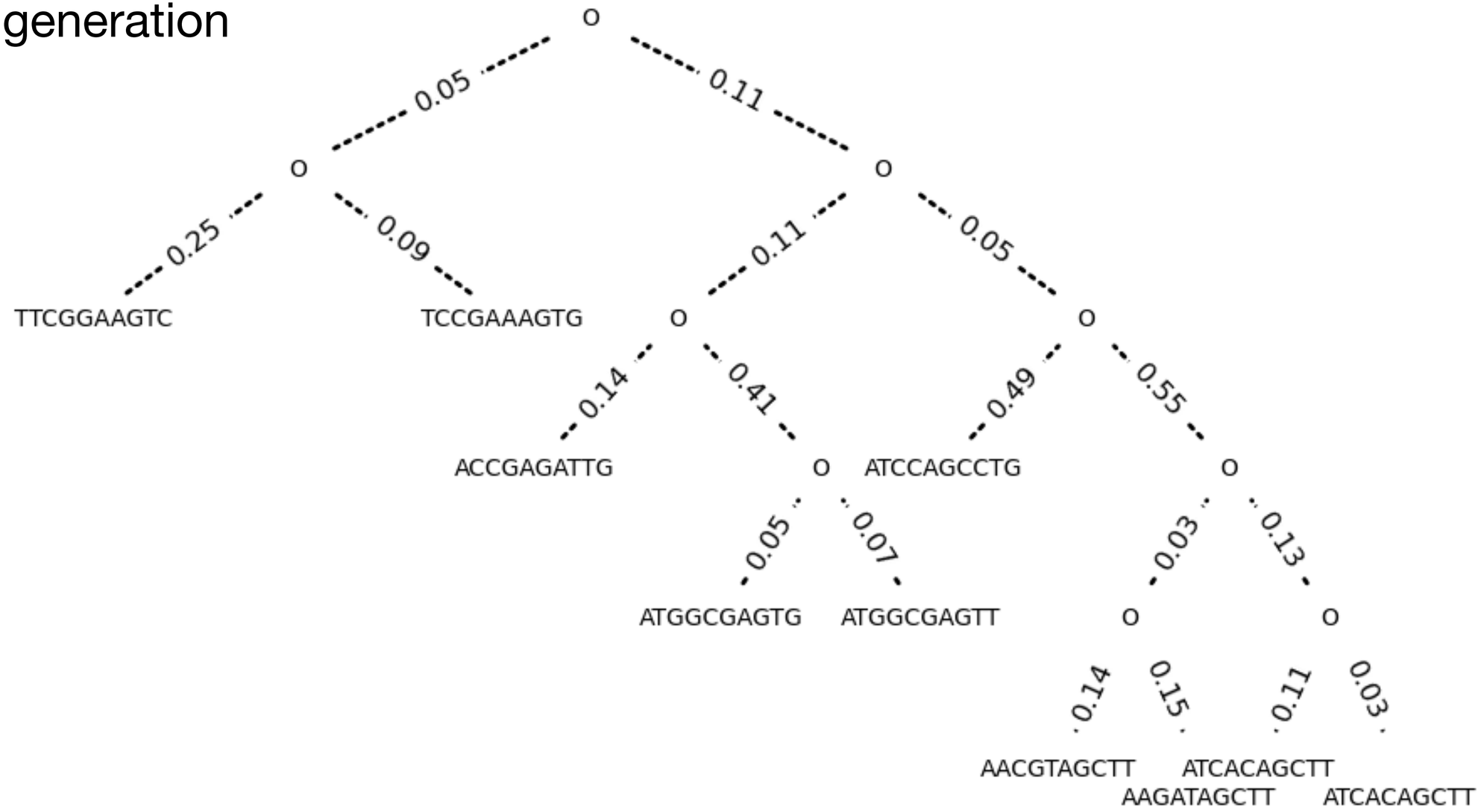


Case study

More complex scenario $\rightarrow n = 10$ leaves, with $m = 10$ sites per sequence

- Synthetic data generation ($\delta_t = 0.01$):

  ➡ $\langle t \rangle \approx 0.16$



Case study

More complex scenario $\rightarrow n = 10$ leaves, with $m = 10$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 1, N_{SPR} = 1$

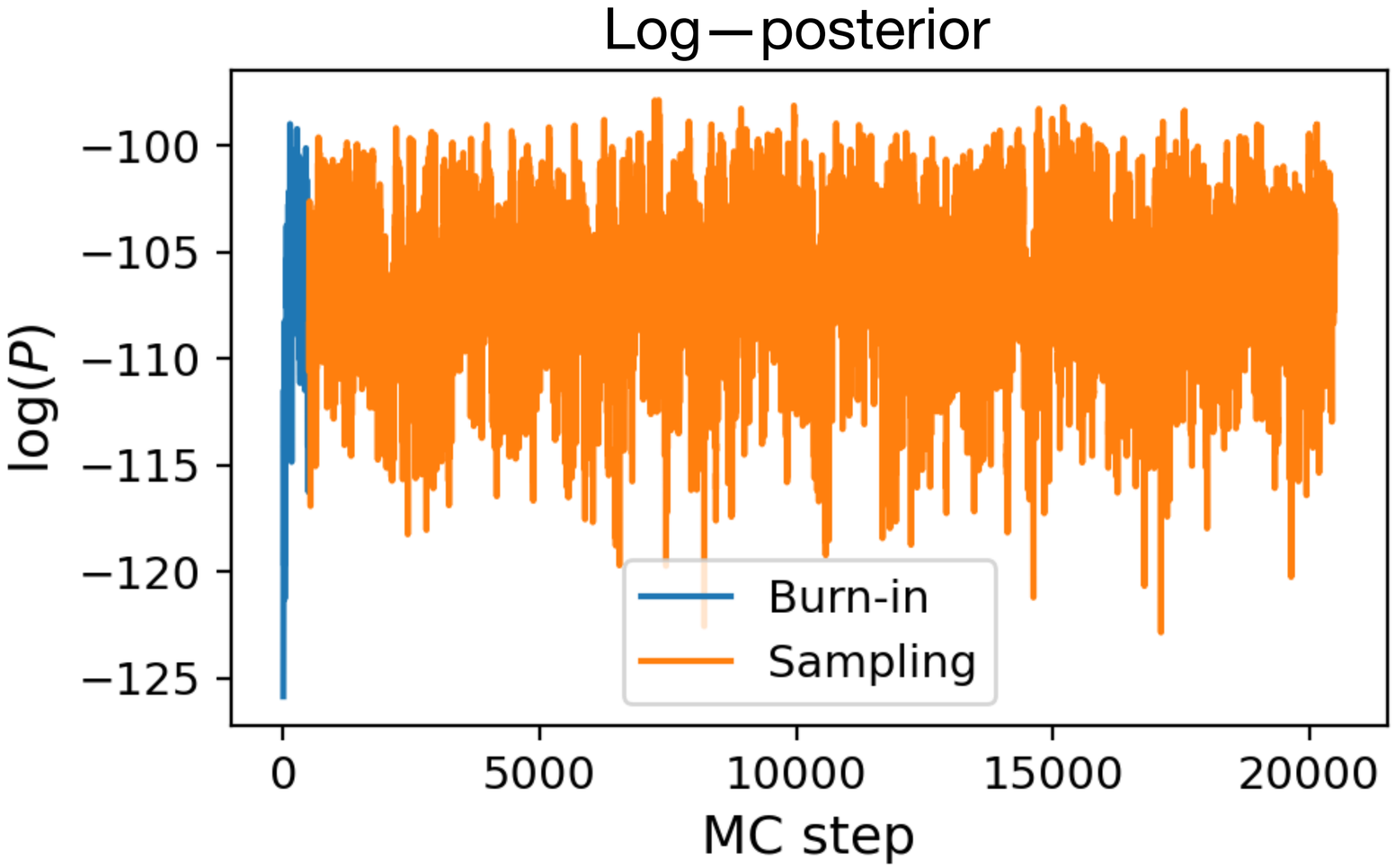
Execution time ≈ 4.7 min

Case study

More complex scenario → $n = 10$ leaves, with $m = 10$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 1, N_{SPR} = 1$

Execution time ≈ 4.7 min

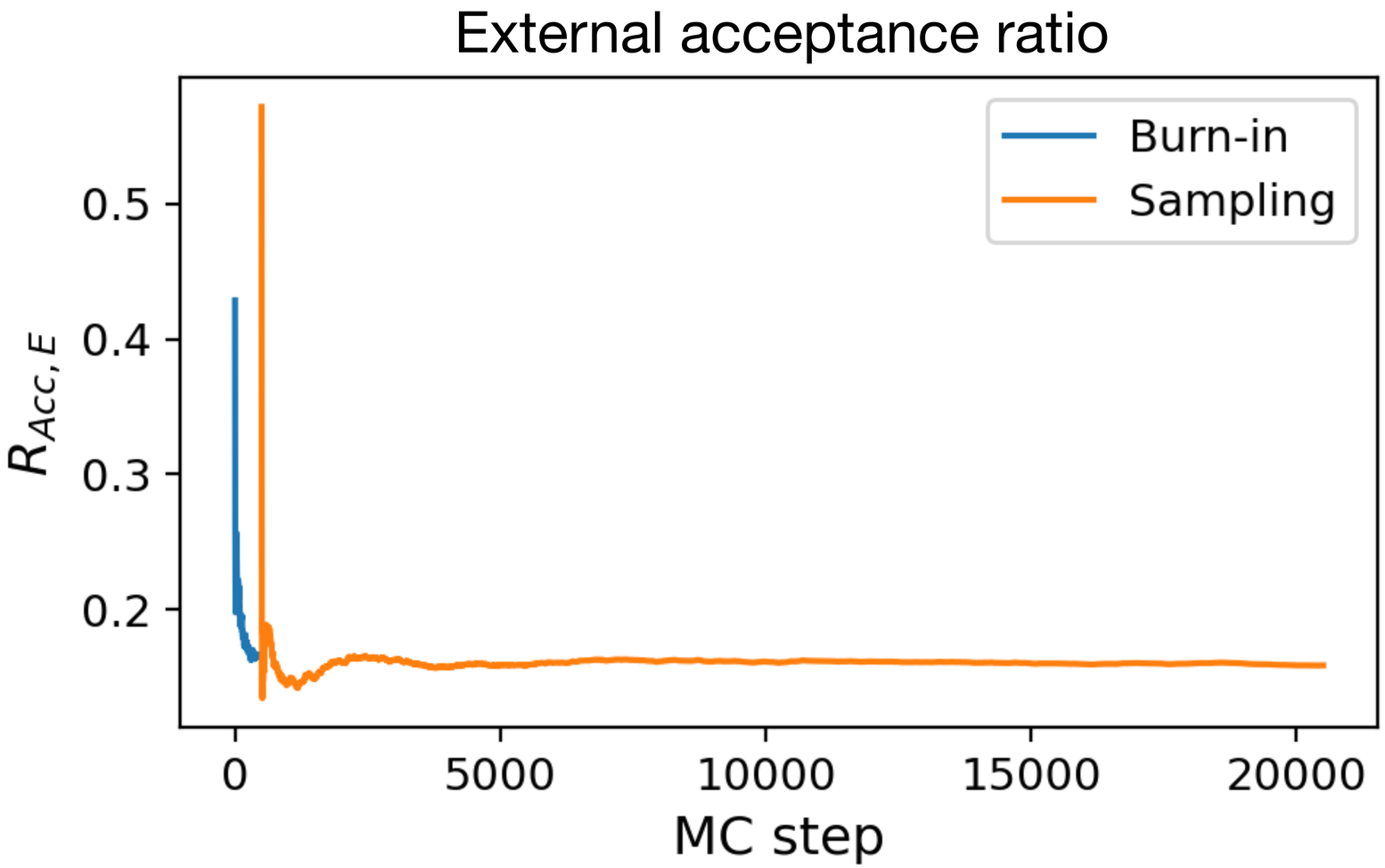


Case study

More complex scenario → $n = 10$ leaves, with $m = 10$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 1, N_{SPR} = 1$

Execution time ≈ 4.7 min

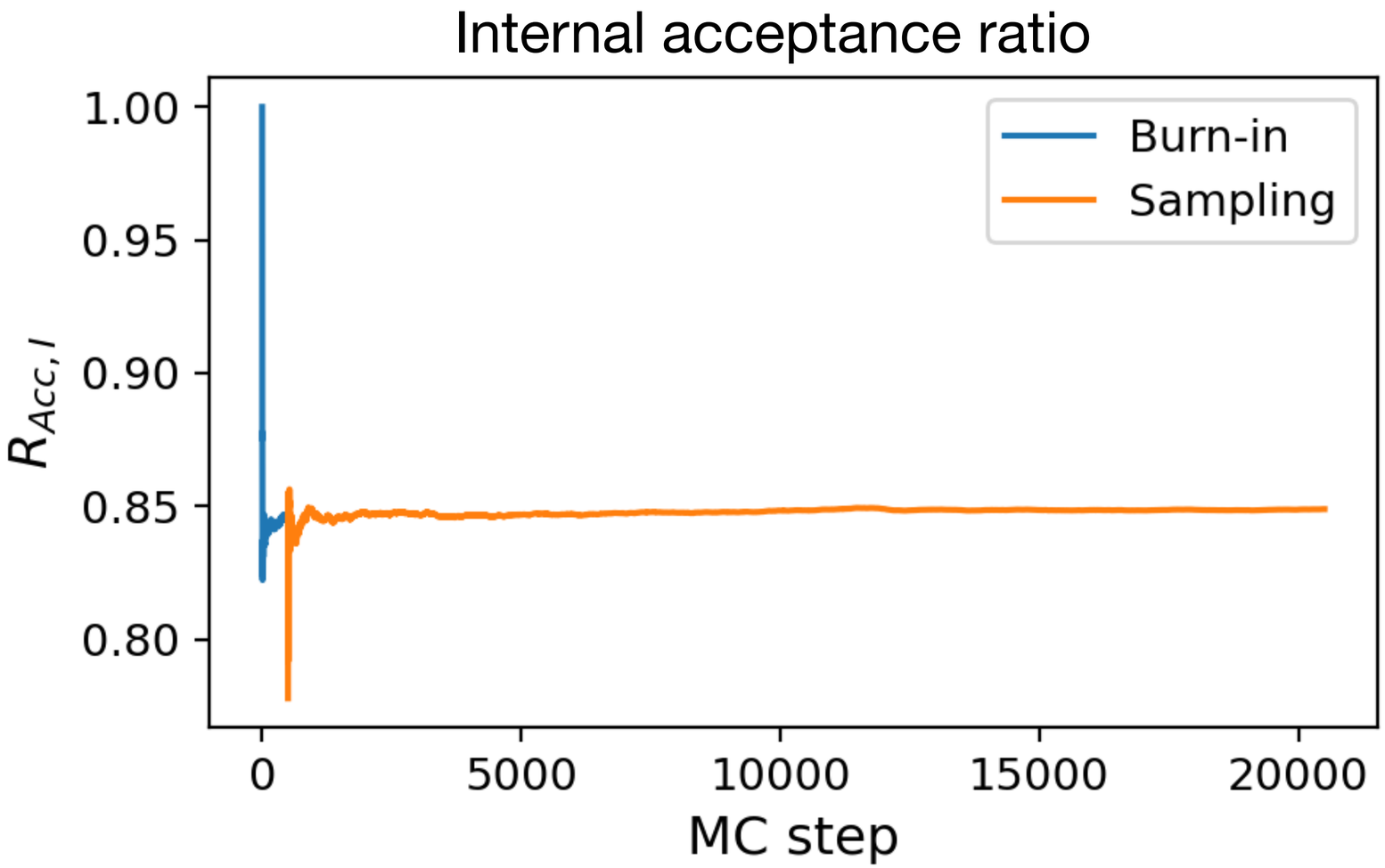


Case study

More complex scenario → $n = 10$ leaves, with $m = 10$ sites per sequence

- MCMC inference:
  - 5×10^2 burn-in steps
  - 2×10^4 sampling steps
  - $\gamma_0 = 1, N_{SPR} = 1$

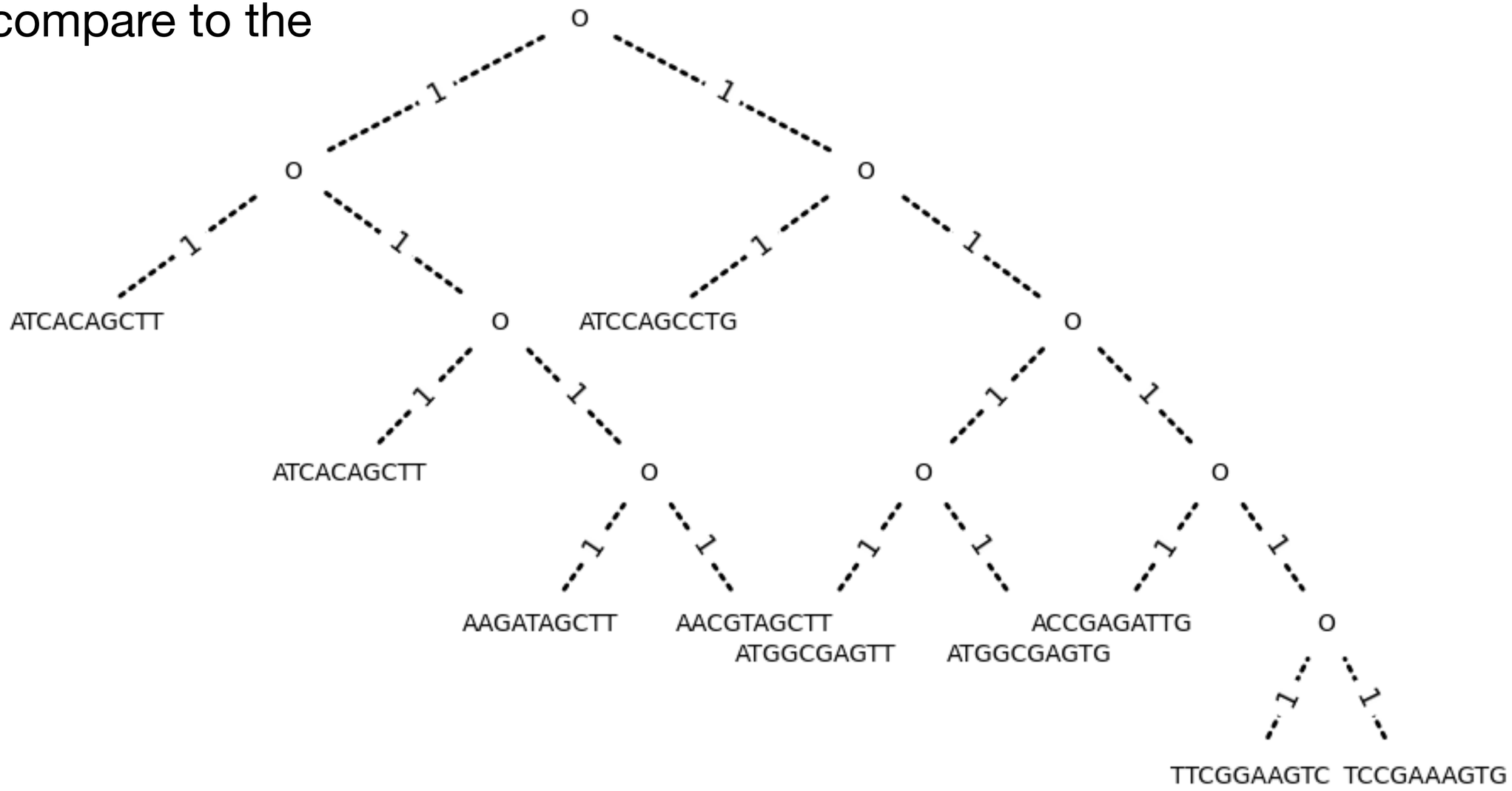
Execution time ≈ 4.7 min



Case study

Maximum a posteriori (MAP) tree → Sampled topology with highest posterior

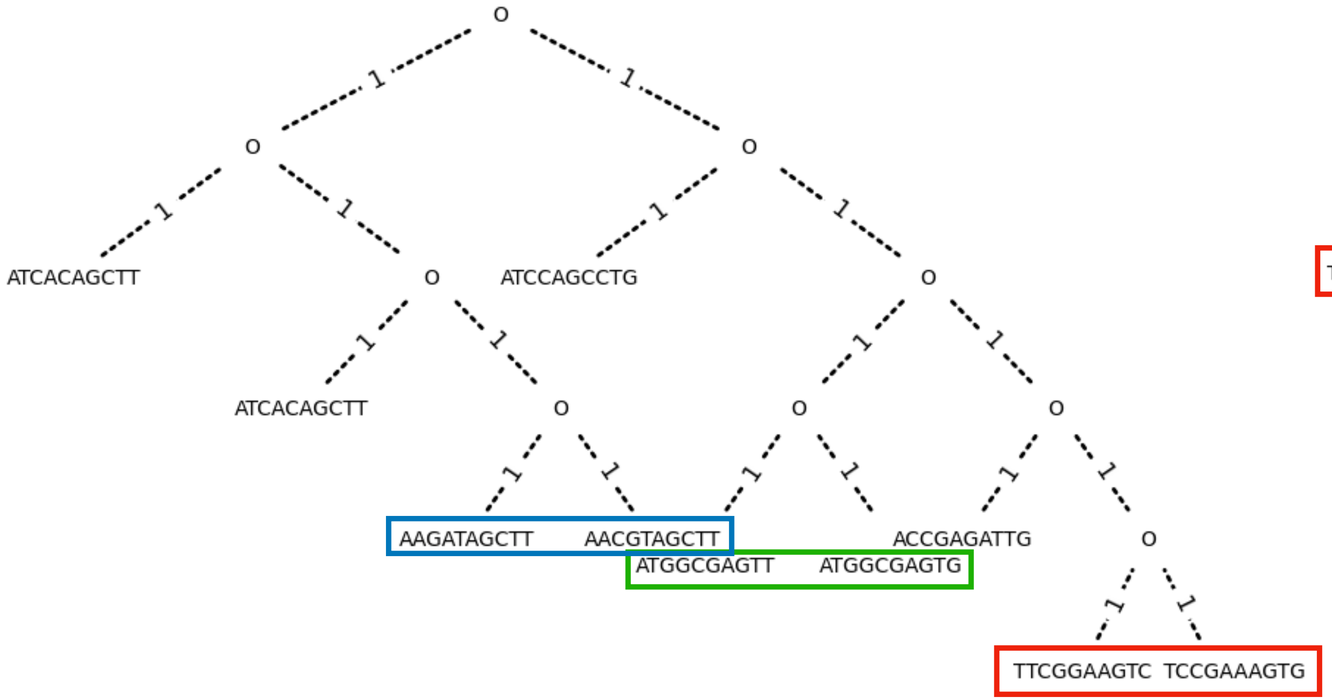
- How does it compare to the original tree?



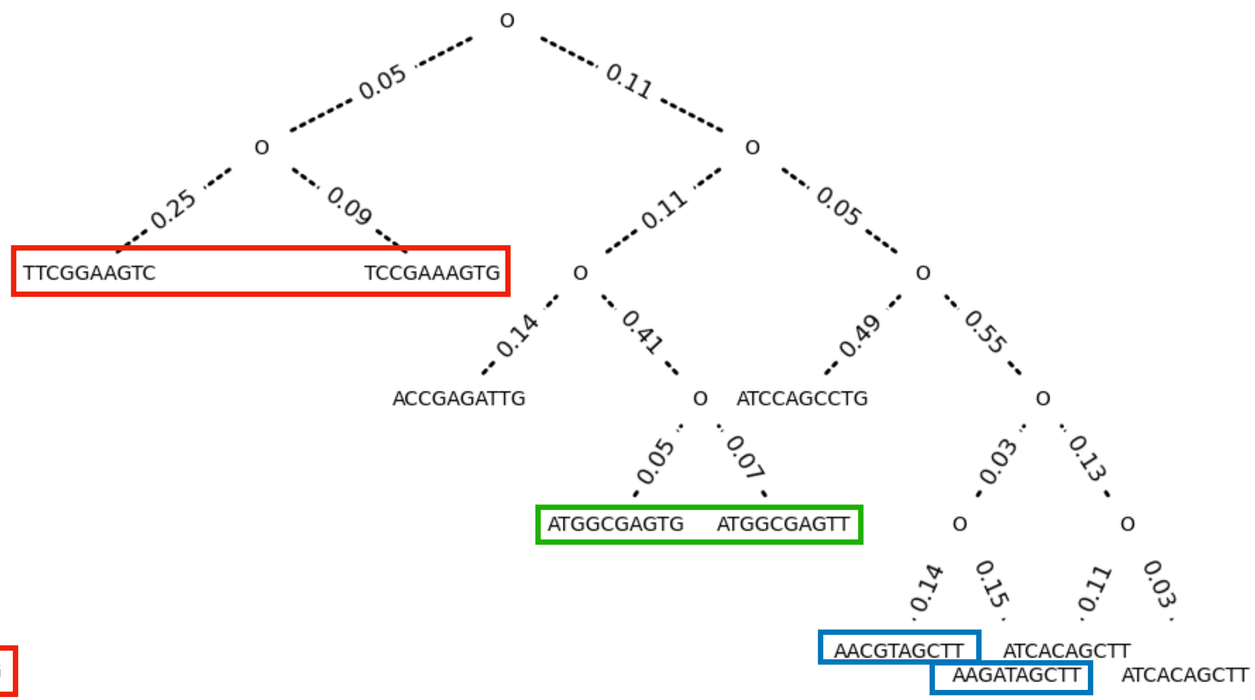
Case study

Maximum a posteriori (MAP) tree → Sampled topology with highest posterior

MAP



Original

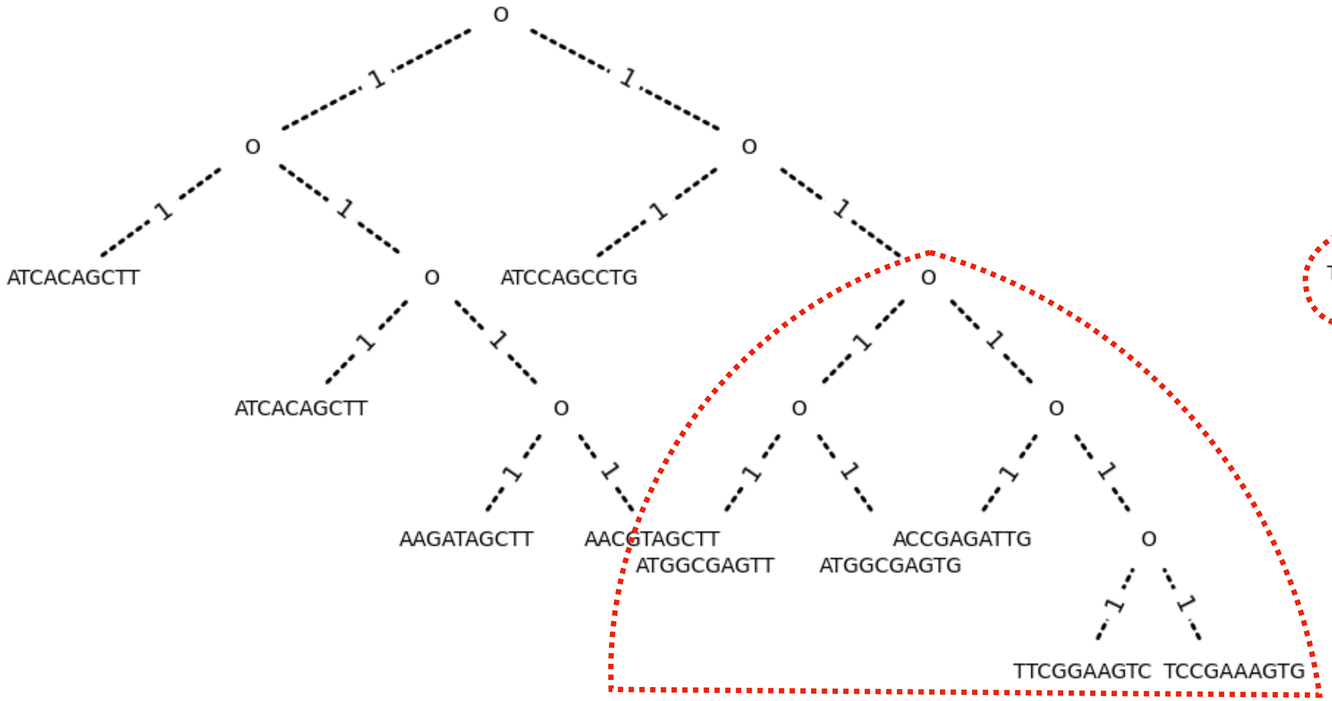


Ranking by $\log(P)$: 2596 out of 3797

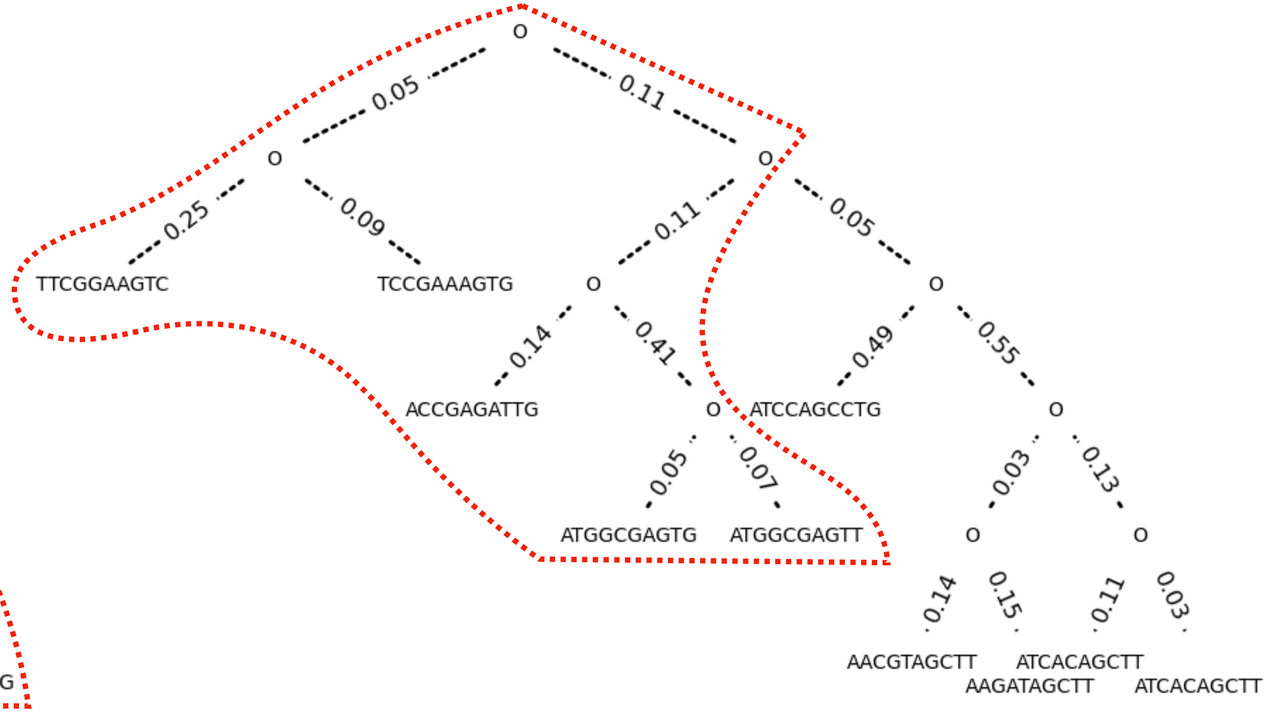
Case study

Maximum a posteriori (MAP) tree → Sampled topology with highest posterior

MAP



Original



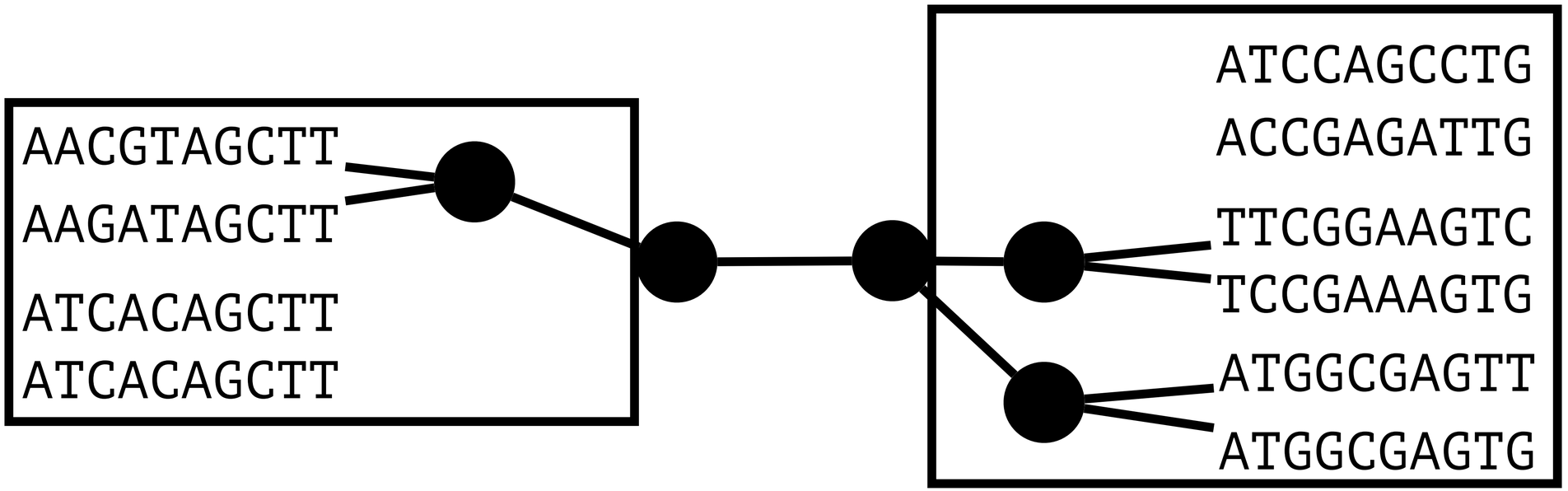
Ranking by $\log(P)$: 2596 out of 3797

Case study

Splits $\rightarrow$ any partition of X into two non—empty disjoint subsets

- Majority—rule consensus: 4 out of 4 consensus splits present in the original tree

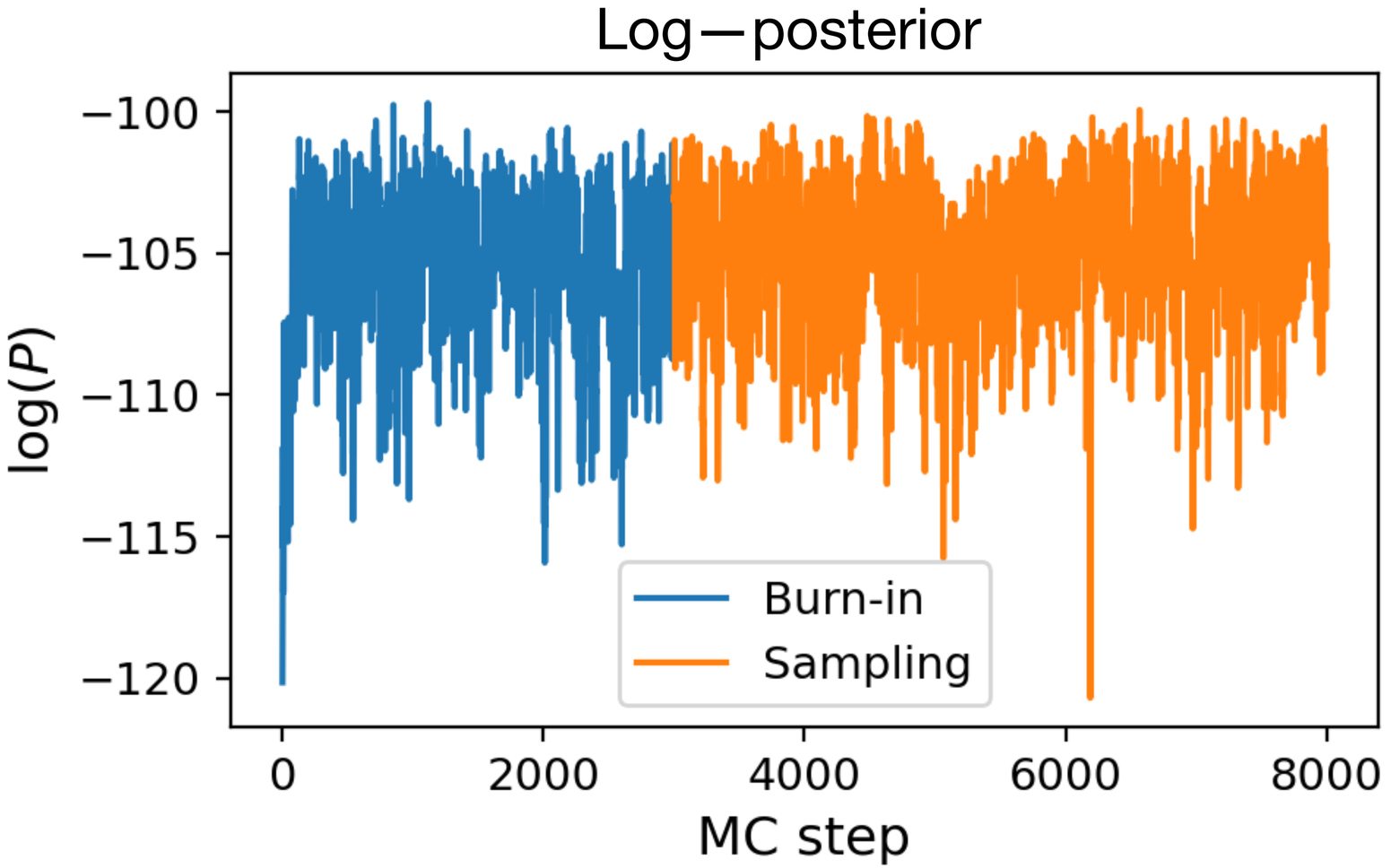
Consensus tree



Case study

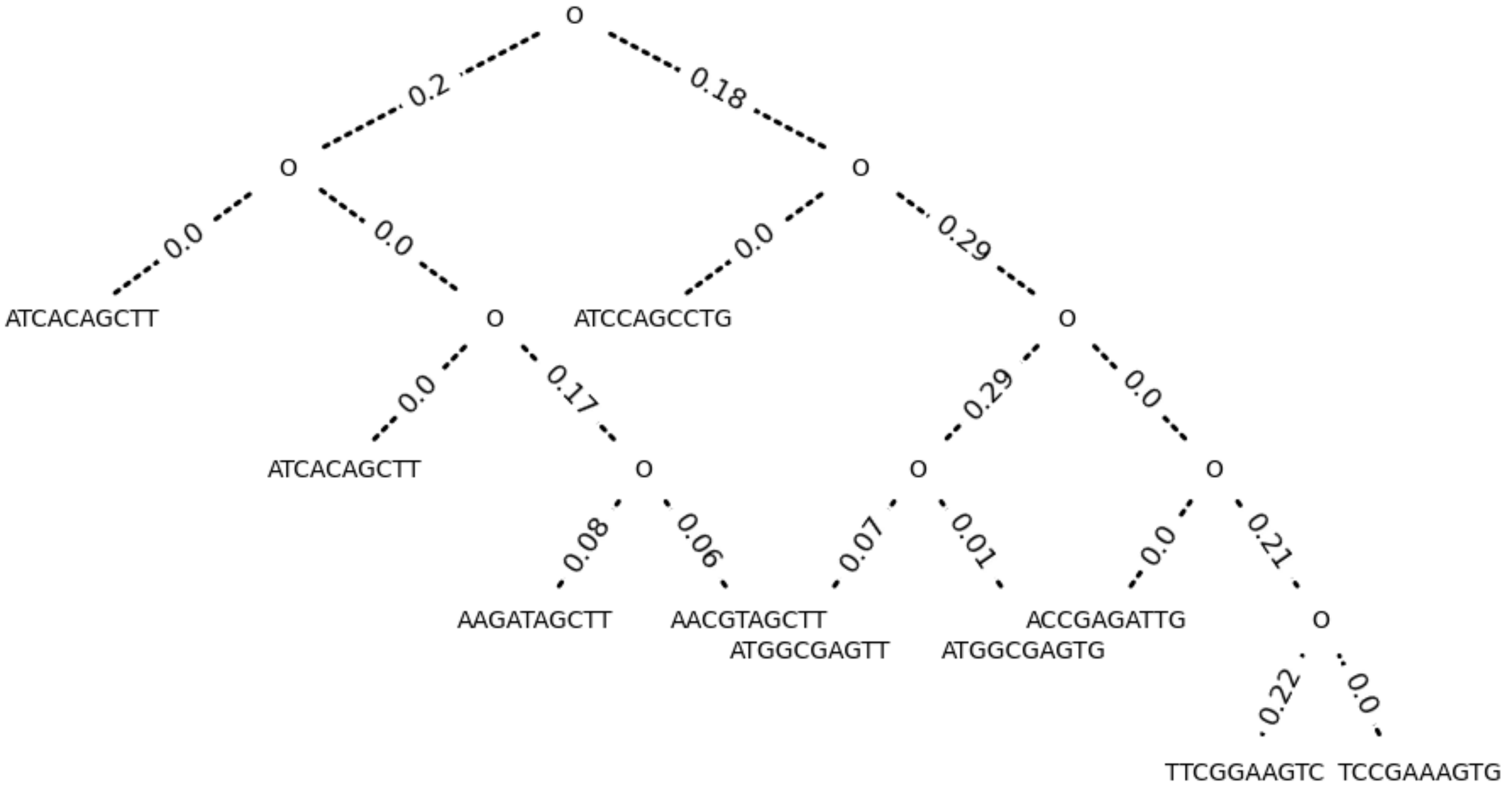
Inference of edge lengths for the MAP tree → Sample with *internal* MC steps

- MCMC inference:
  - 3×10^3 burn-in steps
  - 5×10^3 sampling steps
  - $\gamma_0 = 1$
- Use 1 every 10 samples to reduce correlation.



Case study

Inference of edge lengths for the MAP tree → Sample with *internal* MC steps



CONCLUSION

Development highlights

- Developed a method to sample rooted tree topologies in a non-uniform way, but avoiding biases due to isomorphisms.

Development highlights

- Developed a method to sample rooted tree topologies in a non-uniform way, but avoiding biases due to isomorphisms.
- Performed Monte Carlo steps in an alternating manner, first proposing a new topology (*external* step) and then proposing new edge lengths (*internal* step).

Development highlights

- Developed a method to sample rooted tree topologies in a non-uniform way, but avoiding biases due to isomorphisms.
- Performed Monte Carlo steps in an alternating manner, first proposing a new topology (*external* step) and then proposing new edge lengths (*internal* step).
- Derived a relationship between the overlap between two sequences and the maximum—likelihood edge length

  ➡ From this, derived an upper bound for the mean edge length in data generation, to ensure that the original phylogenetic tree can be inferred from the sequences

Development highlights

- Developed a method to sample rooted tree topologies in a non-uniform way, but avoiding biases due to isomorphisms.
- Performed Monte Carlo steps in an alternating manner, first proposing a new topology (*external* step) and then proposing new edge lengths (*internal* step).
- Derived a relationship between the overlap between two sequences and the maximum—likelihood edge length

  ➡ From this, derived an upper bound for the mean edge length in data generation, to ensure that the original phylogenetic tree can be inferred from the sequences

Achieved goals

- Simulated phylogenetic data using the Jukes-Cantor sequence evolution model.
- Performed inference on the synthetic data using the Markov Chain Monte Carlo approach → able to partially reconstruct the original phylogenetic relationships.