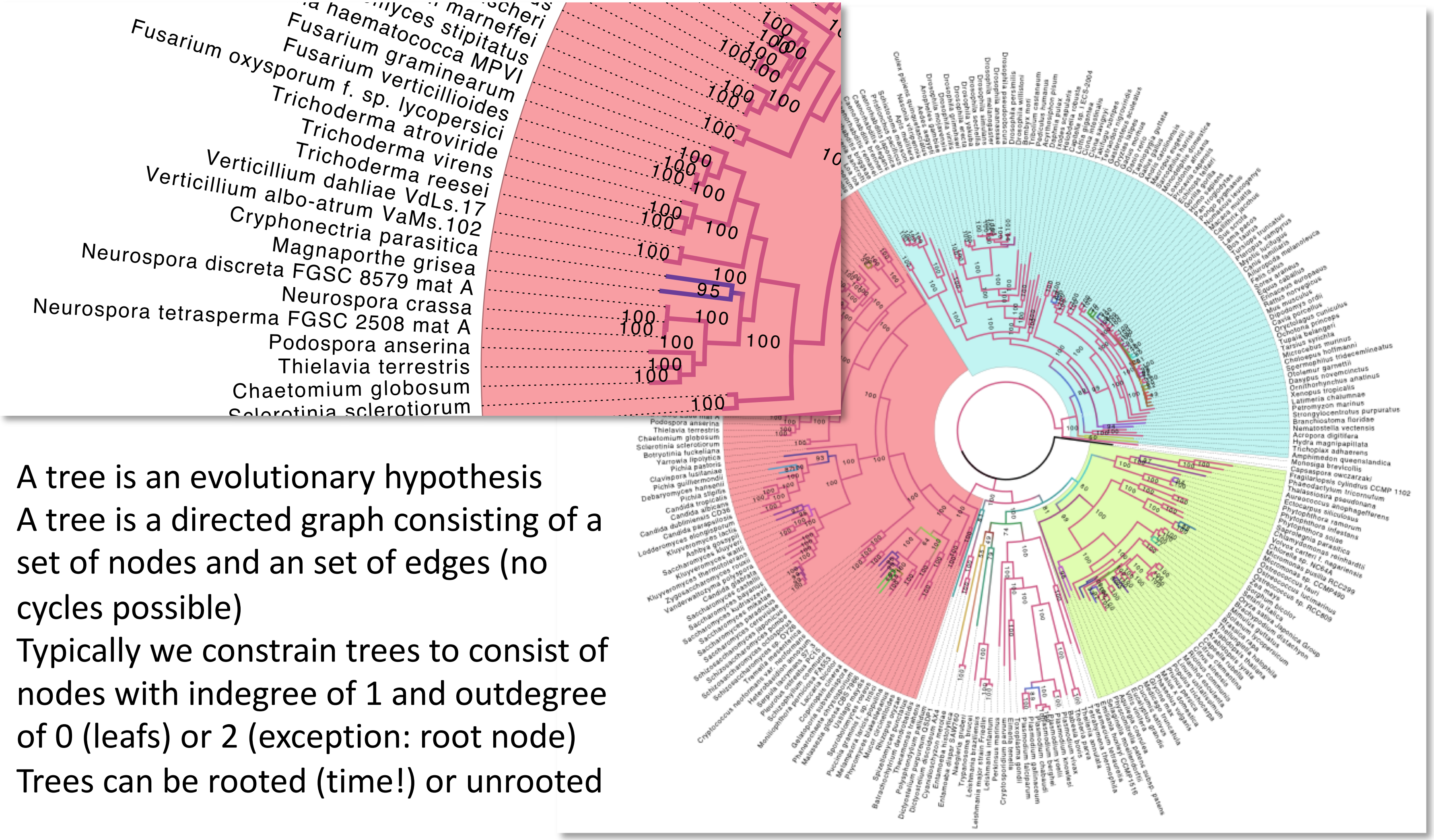
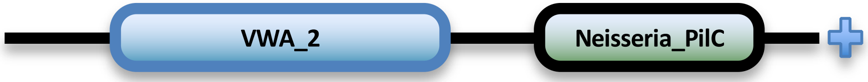
Algorithms in Sequence Analysis 10
Phylogeny Reconstruction

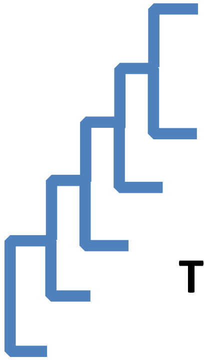


- A tree is an evolutionary hypothesis
- A tree is a directed graph consisting of a set of nodes and an set of edges (no cycles possible)
- Typically we constrain trees to consist of nodes with indegree of 1 and outdegree of 0 (leafs) or 2 (exception: root node)
- Trees can be rooted (time!) or unrooted

Revisiting simulating protein sequence evolution

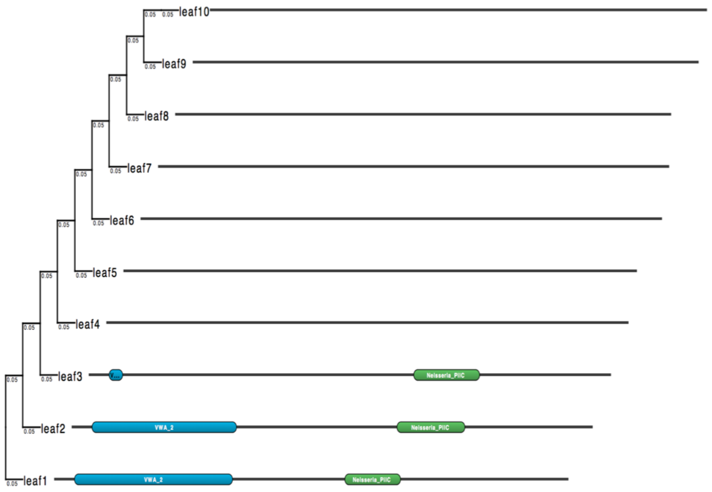


Root sequence



Tree

- Genereate a benchmark data set
- Predict the evolutionary behavior of a sequence[1]

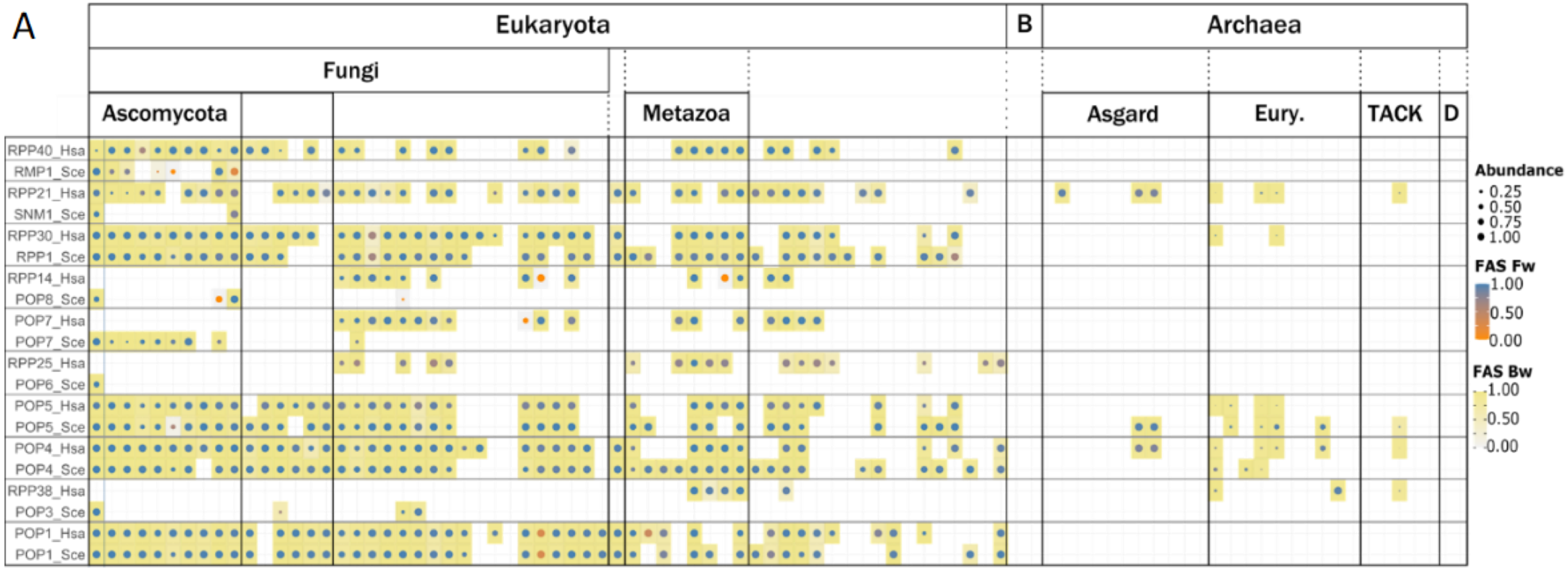


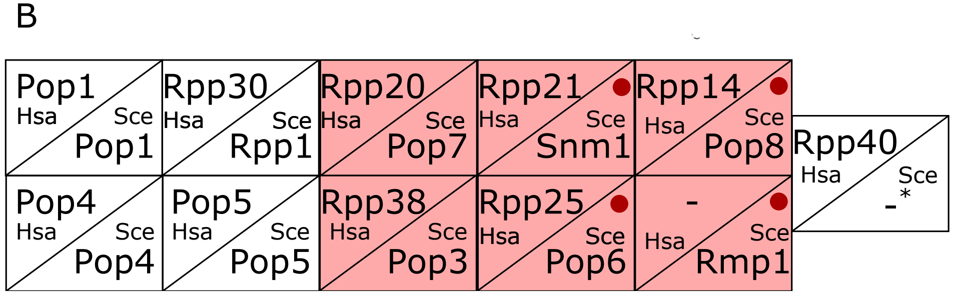
Evolved sequences as leafs of tree

---

[1] Jain et al. (2019) GBE 11:531-545

Why are functionally equivalent proteins not identified as homologs? The yeast and human RNase MRP complexes involved in rRNA processing





Pop1 Hsa / Sce Pop1
Rpp30 Hsa / Sce Rpp1
Rpp20 Hsa / Sce Pop7
Rpp21 Hsa / Sce Snm1
Rpp14 Hsa / Sce Pop8
Rpp40 Hsa / Sce -*
Pop4 Hsa / Sce Pop4
Pop5 Hsa / Sce Pop5
Rpp38 Hsa / Sce Pop3
Rpp25 Hsa / Sce Pop6
- Hsa / Sce Rmp1

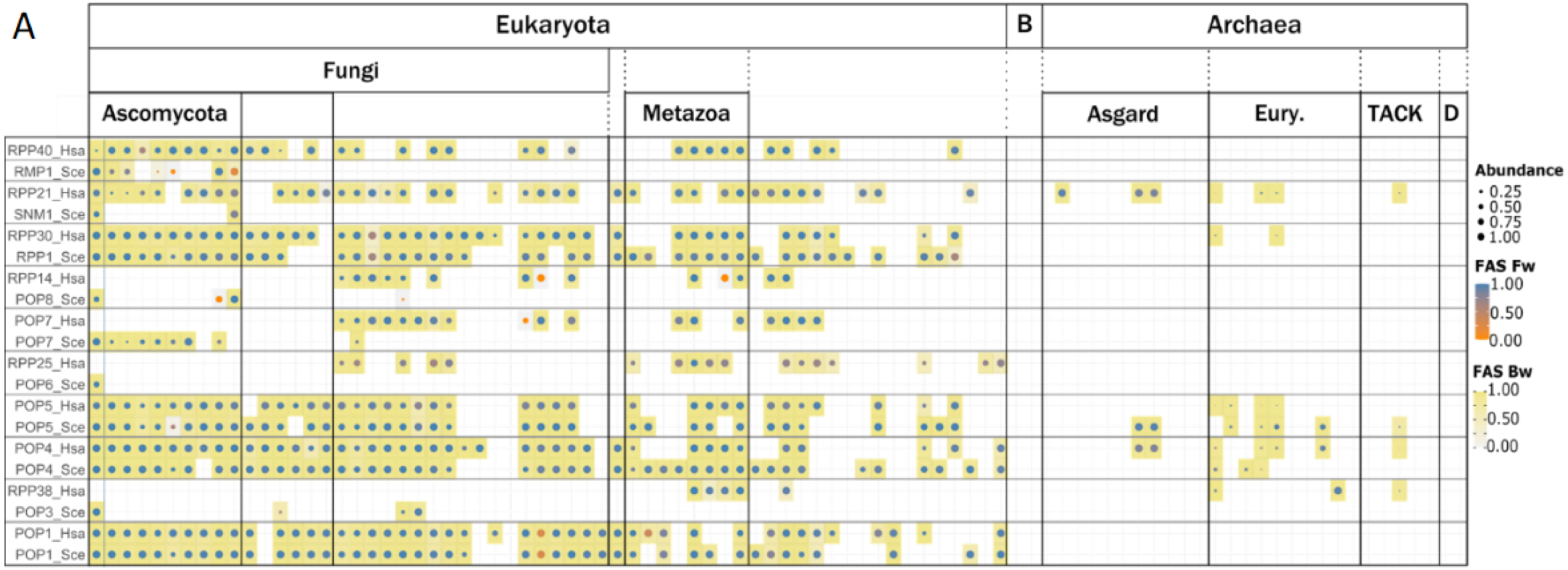
- Each box holds the functionally equivalent human and yeast proteins
- Red boxes indicate proteins that are not identified as orthologs[1]

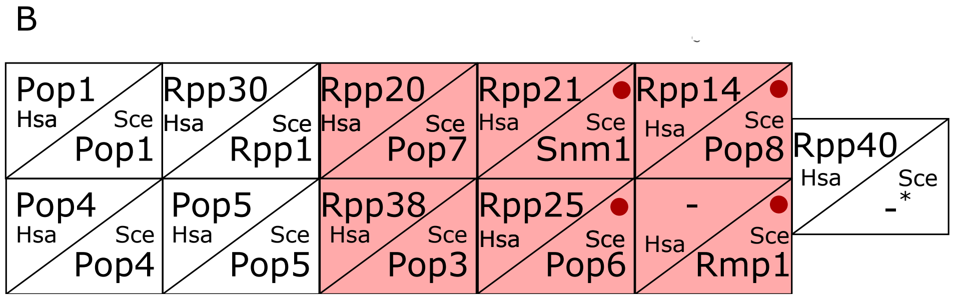
Why??

---

1 Sequences in two species that evolve the same time independent from each other as the species

Why are functionally equivalent proteins not identified as homologs? The yeast and human RNase MRP complexes involved in rRNA processing





Why??

- H1 – the function is taken over by non-homologous proteins
- Proteins evolve too quickly and share no sig. sequence similarity

The simulation of time dependent sequence change using the Gillespie algorithm[1]

Initialization: Initialize the number of evolving positions in the system, event types, event rates, simulation time, and random number generators.

Monte Carlo step: Generate random numbers to determine the next event to occur as well as the time interval. The probability of a given event to be chosen is proportional to its event rate.

Update: Increase the time step by the randomly generated time in Step 2. Update the event rate based on the event that occurred.

Iterate: Go back to Step 2 unless the simulation time has been exceeded.

```
$\Lambda$ <- $\Lambda_{del}$ + $\Lambda_{ins}$ + $\Lambda_{subst}$
$t_{rem}$ = t
$t_w$ ~ Exp($\Lambda$)
while $t_w \leq t_{rem}$ do
    randVar ~ Unif (0,1)
    If randVar $\leq \Lambda_{ins}/\Lambda$ then
        doInsertion()
    else if randVar $\leq$ ($\Lambda$ins + $\Lambda$del) / $\Lambda$ then
        doDeletion()
    else
        doSubstitution()
    end if
    $\Lambda$ = updateEventRate()
    $t_{rem}$ <- $t_{rem}$ - $t_w$
    $t_w$ ~ Exp($\Lambda$)
end while
```

[1] Gillespie, Daniel T. (1977). "Exact Stochastic Simulation of Coupled Chemical Reactions". The Journal of Physical Chemistry. 81 (25): 2340–2361

Modelling the substitution process in a step-wise manner

```
$\Lambda$ <- $\Lambda_{del} + \Lambda_{ins} + \Lambda_{subst}$
$t_{rem}$ = t
$t_w \sim Exp(\Lambda)$
while $t_w \leq t_{rem}$ do
    randVar ~ Unif (0,1)
    If randVar $\leq \Lambda_{ins}/\Lambda$ then
        doInsertion()
    else if randVar $\leq (\Lambda ins + \Lambda del) / \Lambda$ then
        doDeletion()
    else
        doSubstitution()
    end if
    $\Lambda$ = updateEventRate()
    $t_{rem}$ <- $t_{rem} - t_w$
    $t_w \sim Exp(\Lambda)$
end while
```

Total Event rate

Time we still have in front of us

Waiting time for the next event

Individual event rates for insertion, deletion, and **substitution**

Exponential distribution with parameter Λ

Uniform distribution

Computing Event rates
Λ_{del} and Λ_{ins}

Given a sequence of length L drawn at random from the Alphabet A={A,G,C,T} with

frequencies $\pi_i = \frac{1}{4}, \forall i \in \{A,C,G,T\}$

Be λ_{del} the deletion rate per position, and λ_{ins} the insertion rate per position, the we compute the event rates for deletions as

$\Lambda_{\text{del}} = L\lambda_{\text{del}}$

and likewise the event rate for insertions as

$\Lambda_{\text{ins}} = (L+1)\lambda_{\text{ins}}$

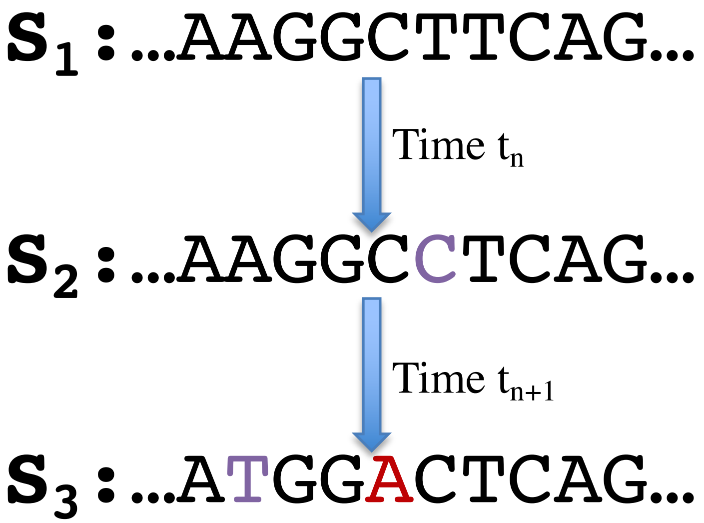
How long should an insertion or a deletion be?
(i.e. what is the probability function modeling the insertion and deletion length distribution?)
And how to compute Λ_{subst}?

Now it is already easy to see why in our simulation algorithm we have to update both event rates after each modification of the sequence as both insertions and deletions modify L.

Computing Event rates

Λ_{subst}

Given a sequence of length l drawn at random from the Alphabet A={A,G,C,T} with frequencies $\pi_i = \frac{1}{4}, \forall i \in \{A,C,G,T\}$



S_1 : ...AAGGCTTCAG...

Time t_n

S_2 : ...AAGGCCTCAG...

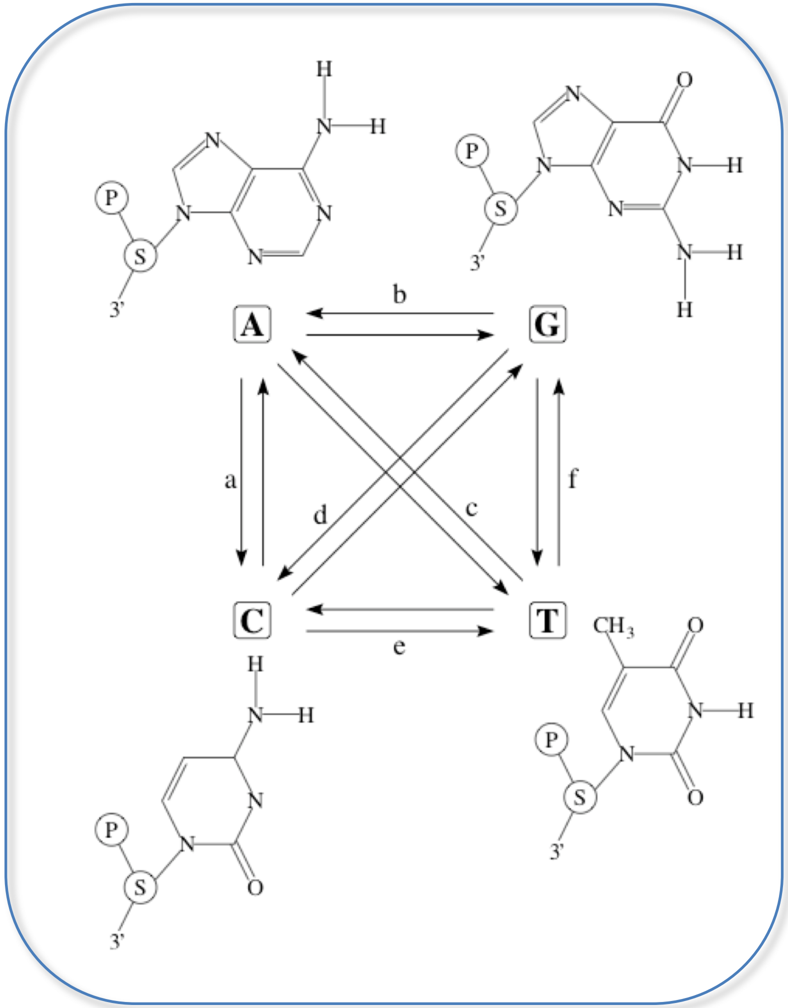
Time t_{n+1}

S_3 : ...ATGGACTCAG...

1. **Markov chain of first order** The evolutionary process has no memory, i.e. sequence S_2 evolves to S_3 in time t_{n+1} independent from S_1
2. **Stationary**
   The frequency π_j of the nucleotides oder amino acids do not change.
3. **time-reversible**

   $\pi_i \cdot q_{ij} = q_{ji} \cdot \pi_j$

The rate matrix *Q* provides the rates for the 12 possible transitions between the nucleotides*. Assuming time reversibility reduces the number of rates to 6.



$$\begin{array}{c} \\ Q = \end{array} \begin{array}{c} \begin{array}{cccc} \mathbf{A} & \mathbf{C} & \mathbf{G} & \mathbf{T} \end{array} \\ \begin{pmatrix} - & a & b & c \\ a & - & d & e \\ b & d & - & f \\ c & e & f & - \end{pmatrix} \end{array}$$

The subsitution rate of nucleotide *i* to nucleotide *j* is provided by the entry q_{ij} in the rate matrix Q.

---

*Die four nucleotides can be considered again as ‘states’ **Rates are **no probabilities,** as they are time-independent!

Computing Event rates
Λ_{subst}

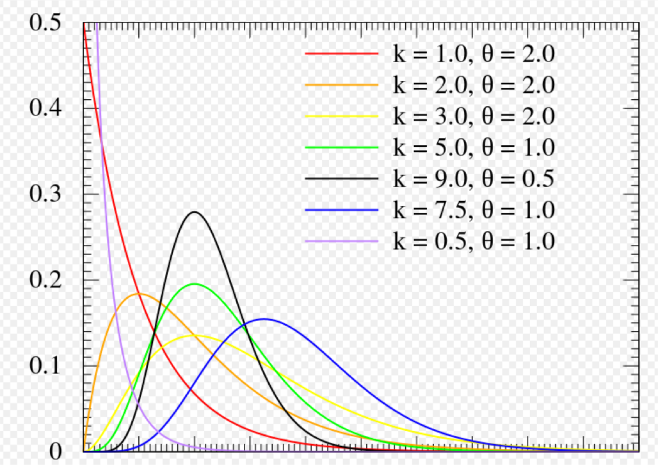
Given a sequence of length *L* drawn at random from the alphabet A={A,G,C,T} with frequencies $\pi_i = \frac{1}{4}, \forall i \in \{A,C,G,T\}$

Let Q be the substitution rate matrix

Then we compute the total rate of substituting base *i* at position *l as*

$$q_{i_l} = \sum_{j \neq i} q_{ij} r_l$$

r_l re-scales time locally. It is drawn from a gamma distribution with a mean of 1 and a shape parameter α



and the total substitution rate Λ_{subst} computes as

$$\Lambda_{subst} = \sum_{l} q_{i_l}$$

How to represent functional constraints in the simulation of the evolutionary process?

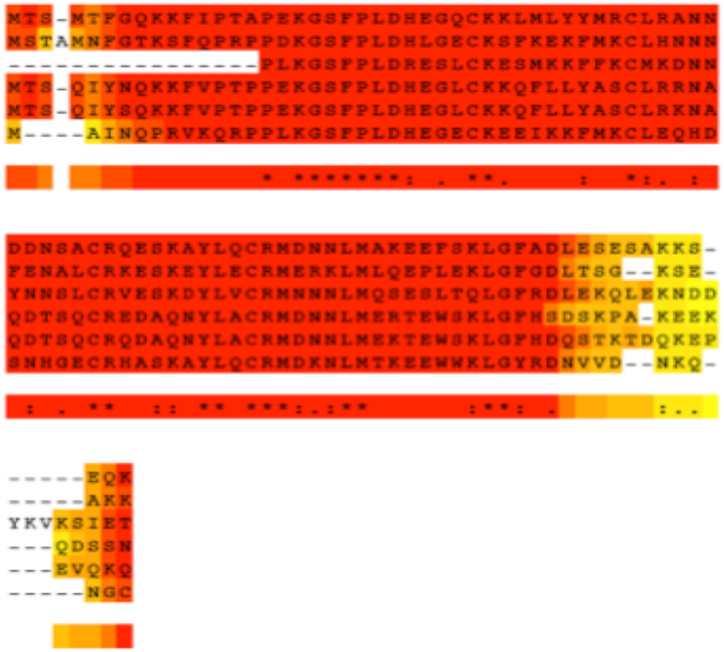
	A	C	G	A	T	G	C	
	q_A	q_C	q_G	q_A	q_T	q_G	q_C	Substitution rate
	r_1	r_2	r_3	r_4	r_5	r_6	r_7	Subst. rate scaling factor
	λ_{del}	λ_{del}	λ_{del}	λ_{del}	λ_{del}	λ_{del}	λ_{del}	Deletion rate
λ_{ins}	λ_{ins}	λ_{ins}	λ_{ins}	λ_{ins}	λ_{ins}	λ_{ins}	λ_{ins}	Insertion rate

Functional constraints can be represented in such a model only via a modification of the position specific rate parameters.
For example, small r_i locally reduce the time that is available for a substitution

How to infer evolutionary constraints on protein sequences?

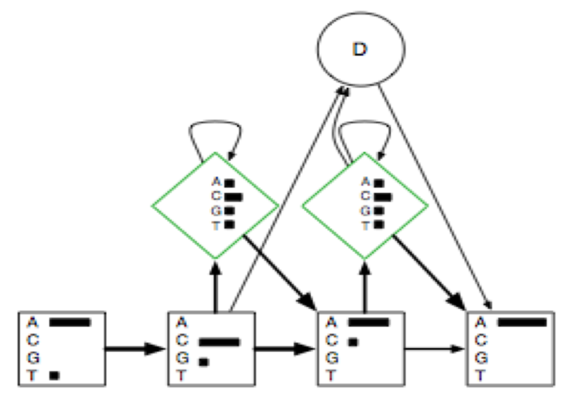
Collect Homologs

Compute MSA



```
MTS-MTFGQKKFIPTAPEKGSFPLDHEGQCKKLMLYYMRCLRANN
MSTAMNFGTKSFQPRPPDKGSFPLDHLGECKSFKEKFMKCLHNNN
----------------PLKGSFPLDRESLCKESMKKFFKCMKDNN
MTS-QIYNQKKFVPTPPEKGSFPLDHEGLCKKQFLLYASCLRRNA
MTS-QIYSQKKFVPTPPEKGSFPLDHEGLCKKQFLLYASCLRKNA
M----AINQPRVKQRPPLKGSFPLDHEGECKEEIKKFMKCLEQHD

                * *******: . **.    :  *:. :

DDNSACRQESKAYLQCRMDNNLMAKEEFSKLGFADLESESAKKS-
FENALCRKESKEYLECRMERKLMLQEPLEKLGFGDLTSG--KSE-
YNNSLCRVESKDYLVCRMNNNLMQSESLTQLGFRDLEKQLEKNDD
QDTSQCREDAQNYLACRMDNNLMERTEWSKLGFHSDSKPA-KEEK
QDTSQCRQDAQNYLACRMDNNLMEKTEWSKLGFHDQSTKTDQKEP
SNHGECRHASKAYLQCRMDKNLMTKEEWWKLGYRDNVVD--NKQ-

 : . **  :: ** ***:.:**      :**: .      :..

-----EQK
-----AKK
YKVKSIET
---QDSSN
---EVQKQ
-----NGC
```

Generate HMM



In a nutshell

- State-specific emission probabilities parameterize substitution rates
- State-specific transition rates parameterize position-specific insertion- and deletion rates

A number of tools exist to simulate the evolution of biological sequences using various evolutionary models

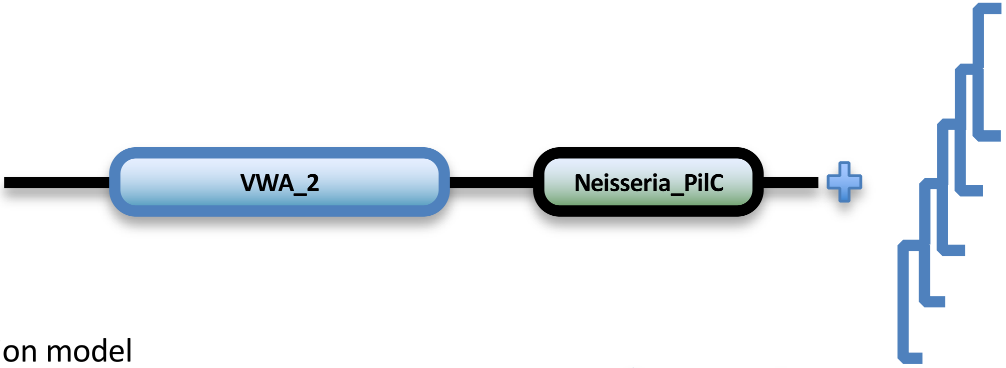
SeqGen (Rambaut and Grassly 1997)
  Event space contains only substitutions
  sites evolve independently and identically
  possibility of randomly assigning rate scaling factors across sites.

Rose (Stoye et al. 1998)
  Event space contains substitutions, insertions and deletions
  Insertions and deletions are randomly placed
  Insertion and deletion lengths are drawn from a single distribution

INDELible (Fletcher and Yang 2009)&Indel-Seq-Gen (Strope et al. 2009)
  facilitate **manual** assignment of model parameters to individual partitions of the data (i.e. local constraints). This helps to modulate the evolutionary process for functional and non-functional regions (‘domains’ vs. ‘linker’)

REvolver (Köstler et al. 2012)
  facilitates an **automated** assignment of constraints on the evolutionary process
  several Indel-Distributions are available
  pHMM guided (Pfam)

Simulating protein sequence evolution: Maintaining evolutionary constraints



General settings

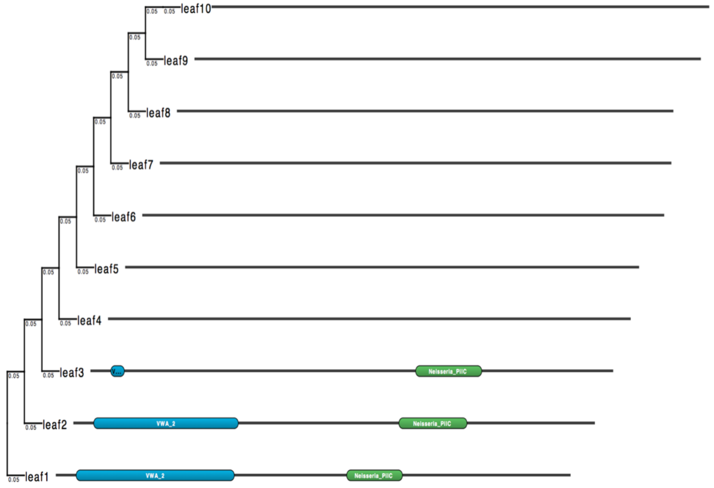
- ➢one substitution model
- ➢site independent evolution
- ➢uniform distribution of indel positions
- ➢one indel length distribution

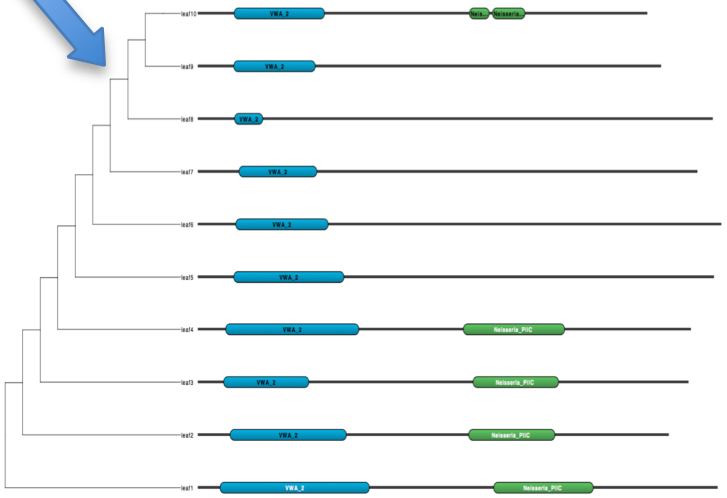
we (typically) get

REvolver[1]

Protein-specific parameterization

- ➢site-specific substitution model
- ➢position-specific indel rates
- ➢position-specific indel length distributions

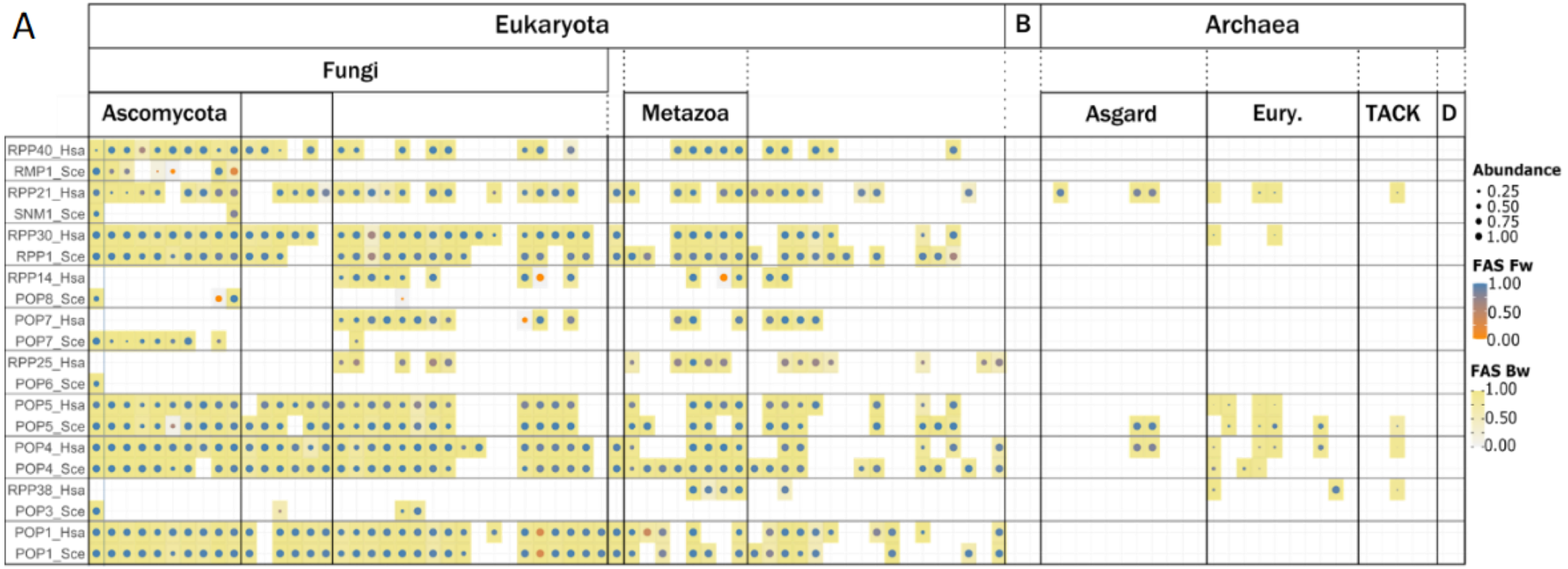


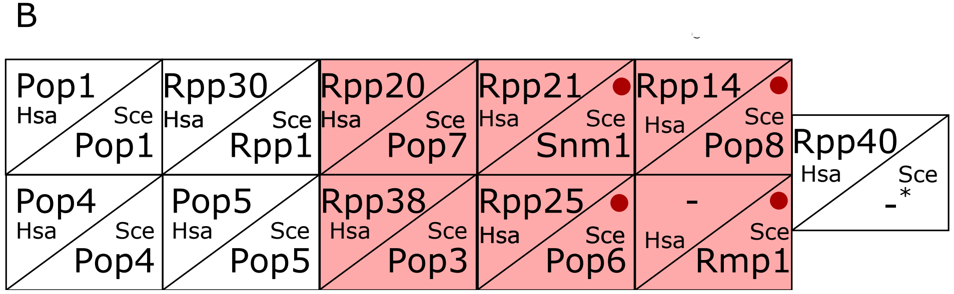


---

[1] Koestler et al. (2012) MBE 29:2133-2145; https://github.com/BIONF/REvolver

Why are functionally equivalent proteins not identified as homologs? The yeast and human RNase MRP complexes involved in rRNA processing

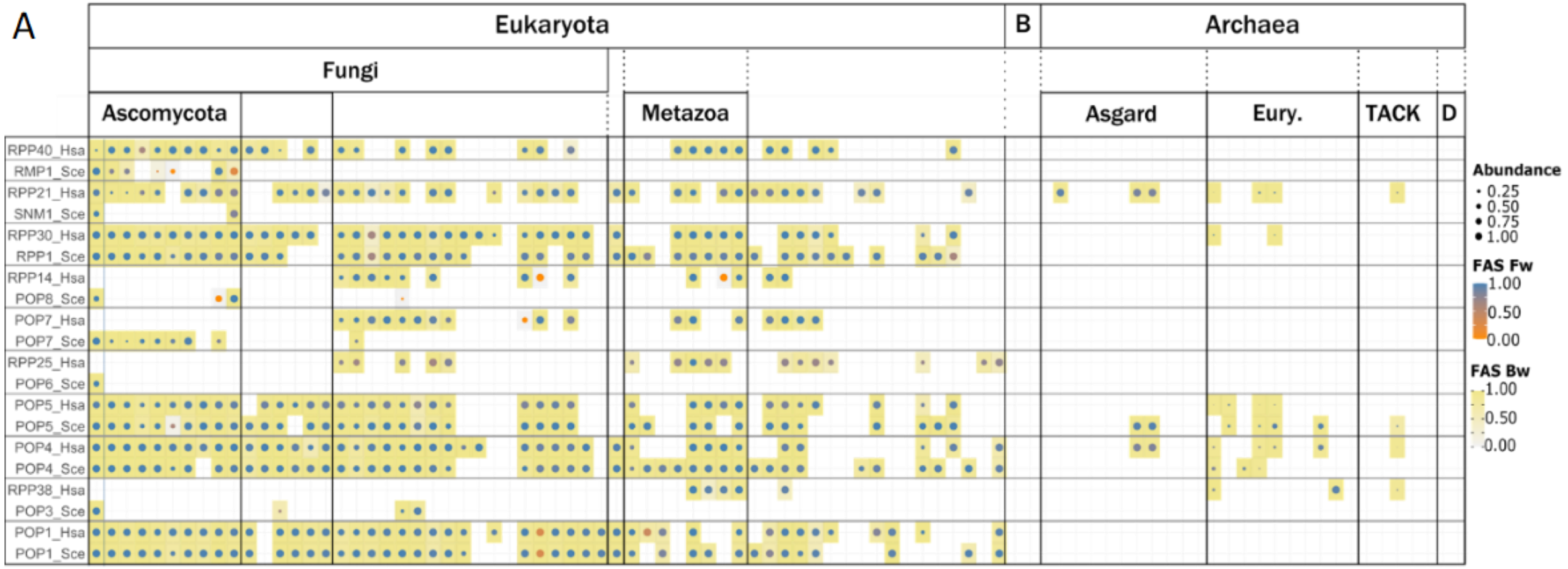


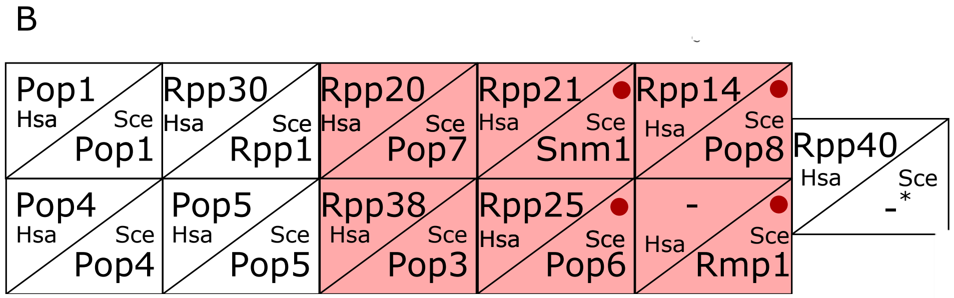


Conclusion

- Simulations shows that Four out of six proteins evolve at rates and patterns that render human and yeast proteins no longer significantly similar

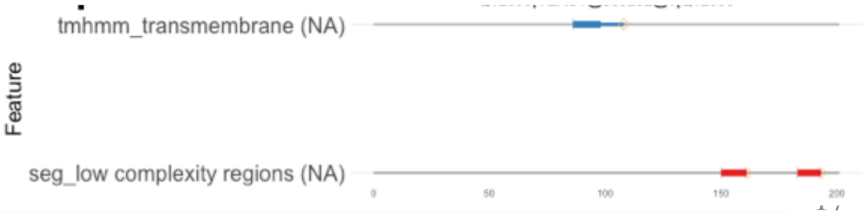
Why are functionally equivalent proteins not identified as homologs? The yeast and human RNase MRP complexes involved in rRNA processing



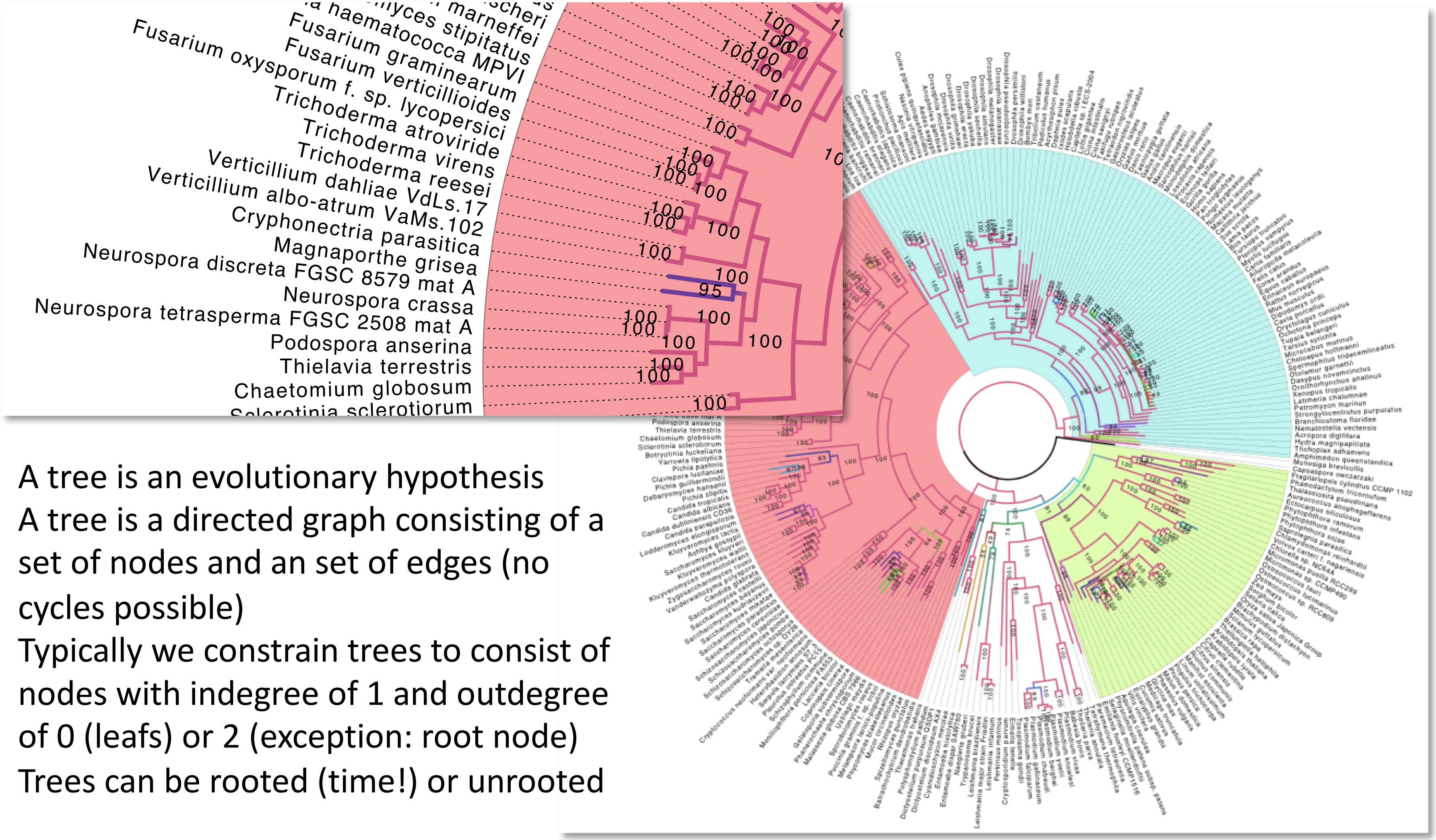


Conclusion

- For example, Rmp1 seems to evolve largely free of constraint



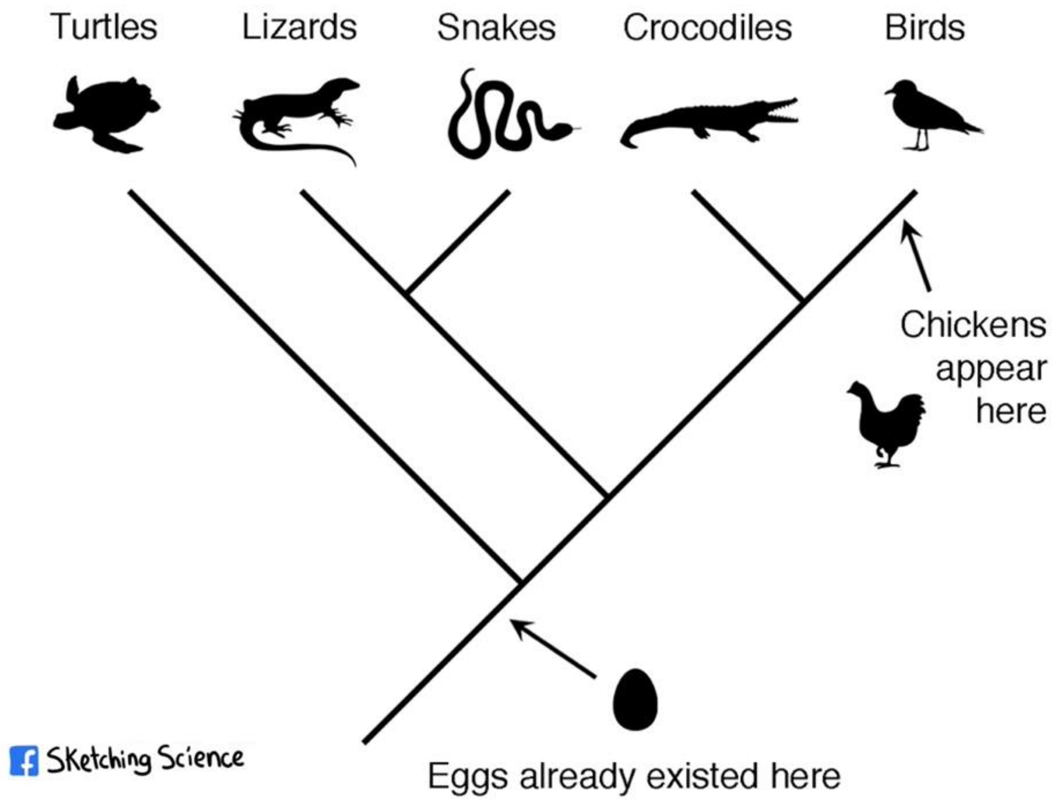
Algorithms in Sequence Analysis 10
Phylogeny Reconstruction



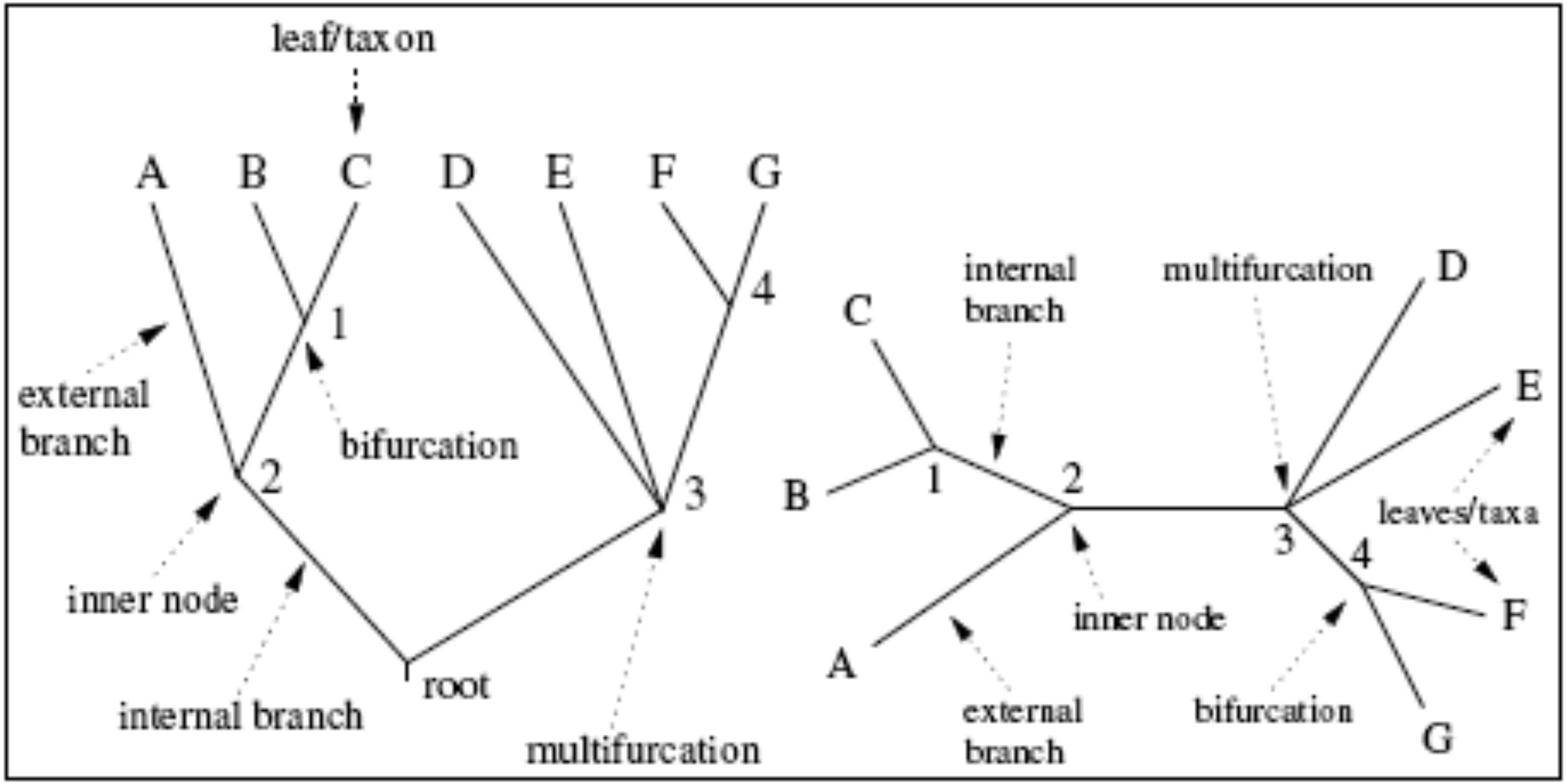
- A tree is an evolutionary hypothesis
- A tree is a directed graph consisting of a set of nodes and an set of edges (no cycles possible)
- Typically we constrain trees to consist of nodes with indegree of 1 and outdegree of 0 (leafs) or 2 (exception: root node)
- Trees can be rooted (time!) or unrooted

What are phylogenetic trees good for?

Which came first, the chicken or the egg?

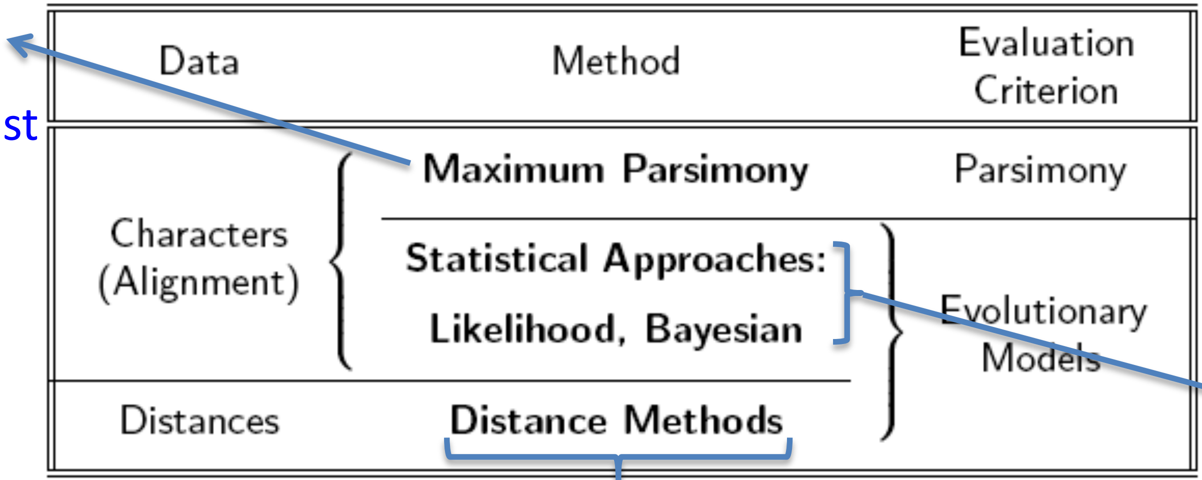


Some notations



Basically, we have three different means to reconstruct phylogenetic trees from sequence data

Find tree that requires the least number of changes



Data	Method	Evaluation Criterion
Characters (Alignment)	**Maximum Parsimony**	Parsimony
	Statistical Approaches: Likelihood, Bayesian	Evolutionary Models
Distances	**Distance Methods**	

Find the tree that most likely gave rise to the data

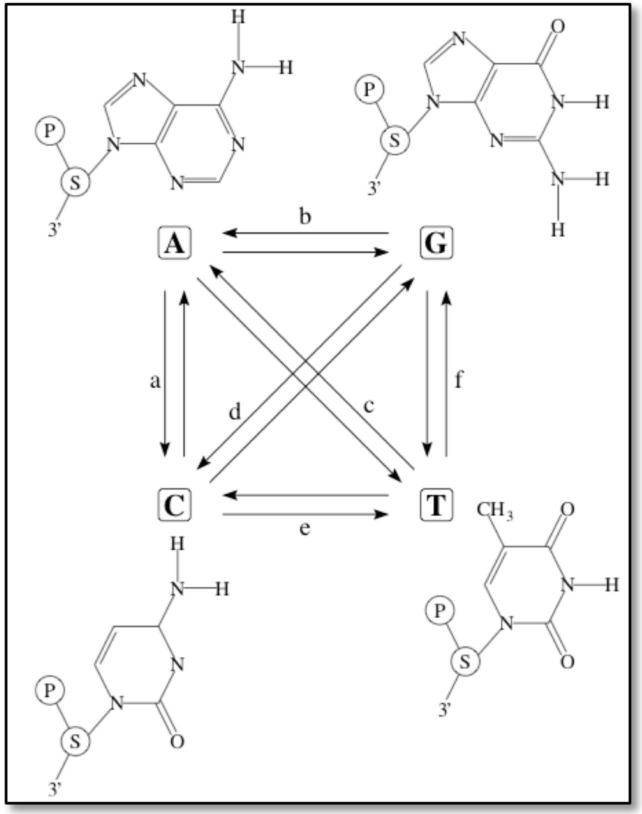
Reconstruct the best fitting tree from a pair-wise distance matrix[1]

---

[1] see Grundlagen der Bioinformatik, Lecture 12

Modelling sequence evolution

Evolutionary models are often described using a substitution rate matrix *Q* and character frequencies Π.



$$\begin{array}{cc} & \begin{array}{cccc} A & C & G & T \end{array} \\ Q = & \begin{pmatrix} - & a & b & c \\ a & - & d & e \\ b & d & - & f \\ c & e & f & - \end{pmatrix} \end{array}$$

$$\Pi = (\pi_A, \pi_C, \pi_G, \pi_T)$$

From *Q* and Π we reconstruct a substitution probability matrix *P* where $P_{ij}(t)$ is the probability of changing *i* to *j* in time *t.*

$$P(t) = e^{Qt}$$

With the likelihood function, we can now compute the likelihood for a time *t* that separates the sequences *S* and *S'*

S: GGTCCTGACAGAAATAAAC

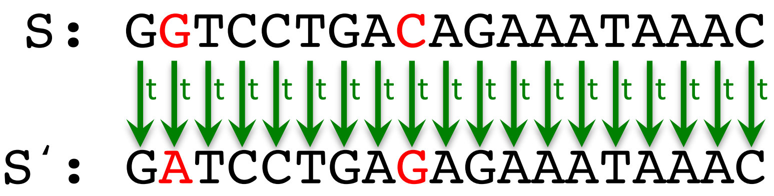
S': GATCCTGAGAGAAATAAAC

$$L(t \mid s \rightarrow s') = \prod_{i=1}^{m}\left(\pi_{s_i} \times P_{s_i s'_i}(t)\right)$$

m: alignment length
S_i: character at position *i* in sequence *S*
S'_i: character at position *i* in sequence *S'*

This value denotes the probability to observe the site pattern (alignment column) at position ***i*** in the alignment.
More precisely, it is the probability that nucleotide $\boldsymbol{S_i}$ has been substituted by nucleotide $\boldsymbol{S'_i}$ after time ***t***.

We can now compute column-wise the likelihood for any time *t* and identify that *t* for which *L(t|S->S')** is maximal over the entire alignment



S: GGTCCTGACAGAAATAAAC

S': GATCCTGAGAGAAATAAAC

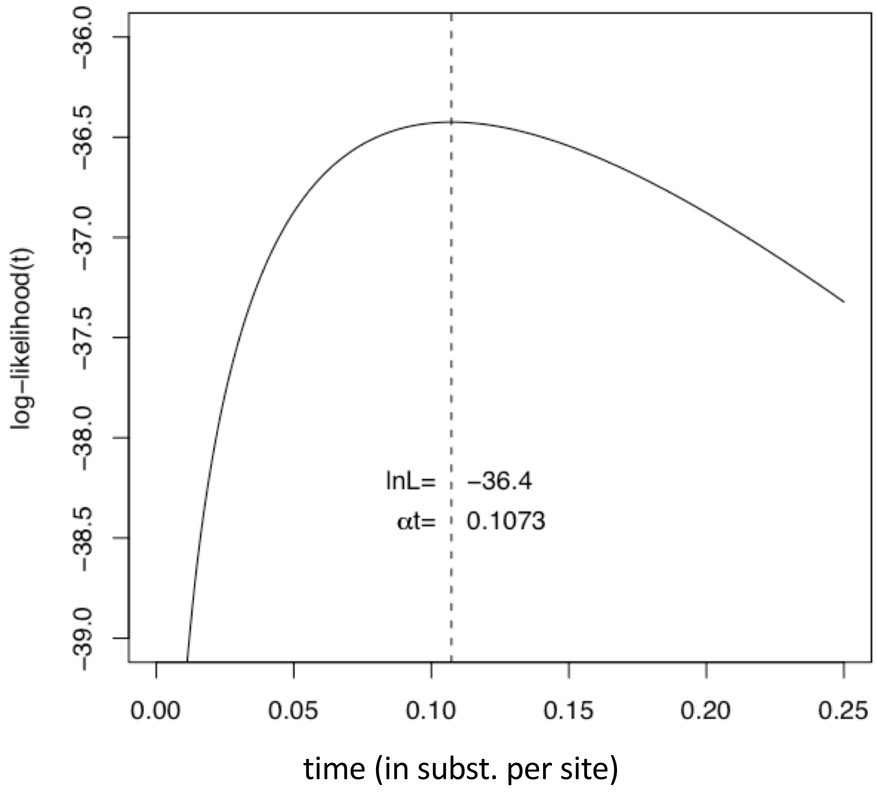
Multiply 'site-likelihoods' across all *m* positions of the alignment

$$L(t \mid s \rightarrow s') = \prod_{i=1}^{m}\left(\pi_{s_i} \times P_{s_i s'_i}(t)\right)$$

Probability (given our model), that S_i was replaced by S'_i in time *t*.

Probability to see the letter at position *i* in the sequence

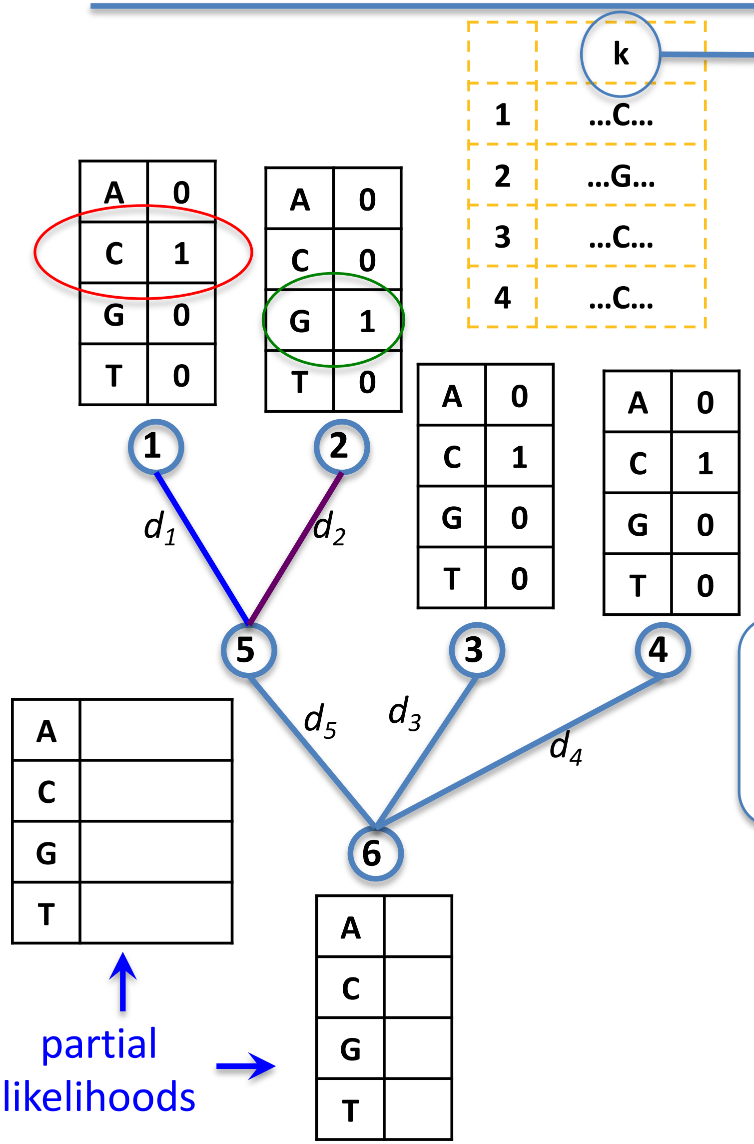
Log-Likelihood surface under JC69



---

*watch out, products of probabilities become small very quckly. We therefore compute typically in log scale, hence the log-likelihoods

Calculating tree likelihoods



	k
1	…C…
2	…G…
3	…C…
4	…C…

A	0
C	1
G	0
T	0

A	0
C	0
G	1
T	0

A	0
C	1
G	0
T	0

A	0
C	1
G	0
T	0

A	
C	
G	
T	

A	
C	
G	
T	

For a single site pattern *k* and a given tree we compute:

$$L_5(i) = [L(C) \times P_{iC}(d_1)] \times [L(G) \times P_{iG}(d_2)], \forall\, i \in \{A,C,G,T\}$$ [1]

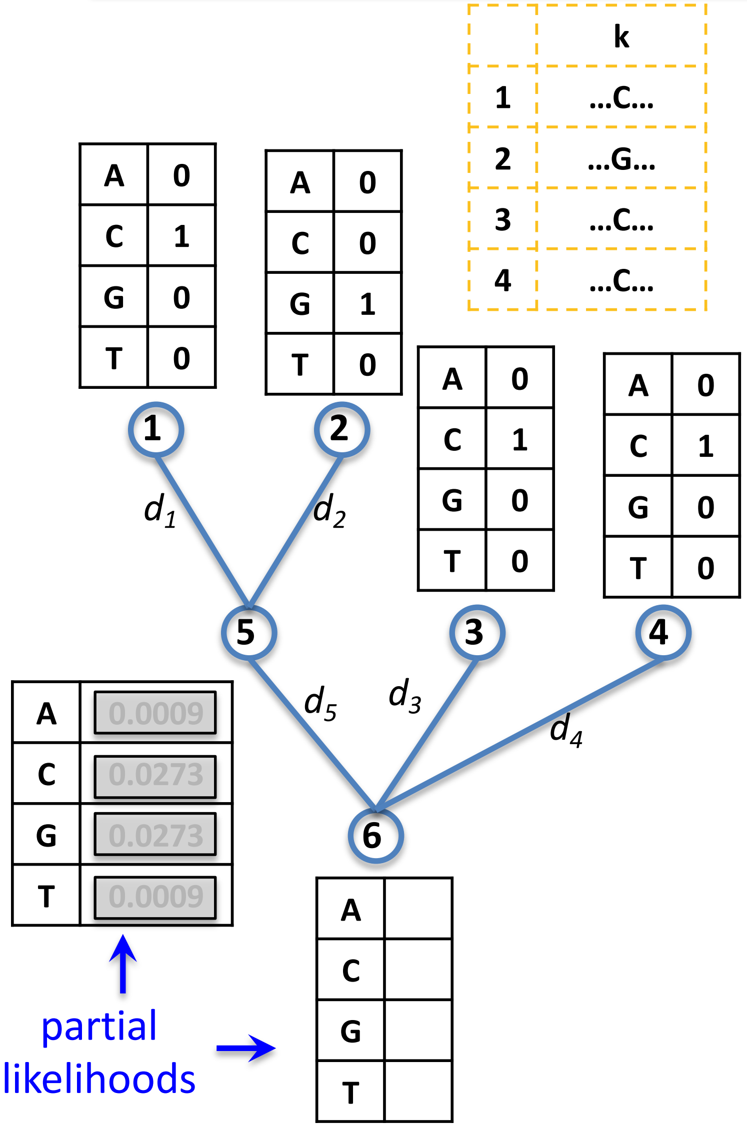
Note, these probabilities do not change as the bases are observed in the data!

Note, these probabilities DO change with *i and d,* and are specified by the substitution model!

with $d_x = 0.1 \ \forall\ x \in \{1,..,5\}$, and $P_{ij}(0.1) = \begin{cases} 0.91 \text{ if } i = j \\ 0.03 \text{ for each } i \neq j \end{cases}$

[1] here you compute the partial likelihood L_5(i) for each ancestral nucleotide *i* in node 5 of the tree, given the data in *k* and the model

Calculating tree likelihoods



	k
1	…C…
2	…G…
3	…C…
4	…C…

A	0
C	1
G	0
T	0

A	0
C	0
G	1
T	0

A	0
C	1
G	0
T	0

A	0
C	1
G	0
T	0

A	0.0009
C	0.0273
G	0.0273
T	0.0009

A	
C	
G	
T	

partial likelihoods

For a single site pattern *k* and a given tree:

$$L_5(i) = [L(C) \times P_{iC}(d_1)] \times [L(G) \times P_{iG}(d_2)], \forall\, i \in \{A,C,G,T\}$$

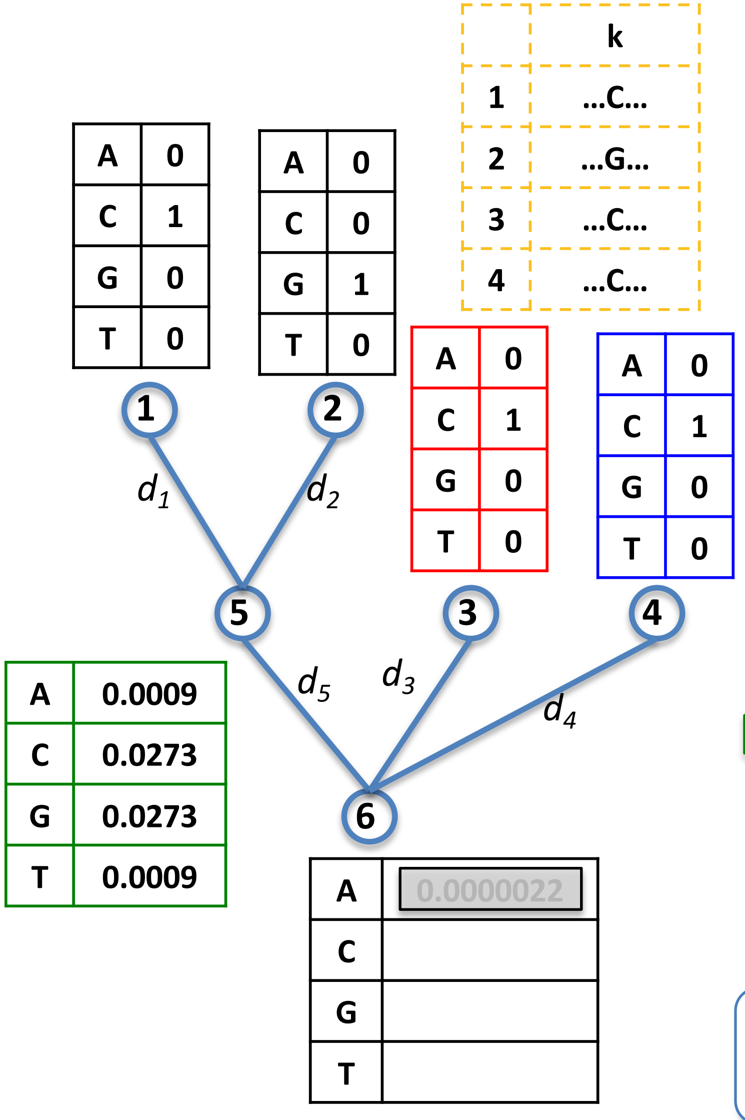
$$L_5(A) = [1 \times P_{AC}(0.1)] \times [1 \times P_{AG}(0.1)]$$
$$= 1 \times 0.03 \times 1 \times 0.03$$
$$= 0.0009$$

$$L_5(C) = [1 \times P_{CC}(0.1)] \times [1 \times P_{CG}(0.1)]$$
$$= 1 \times 0.91 \times 1 \times 0.03$$
$$= 0.0273$$

L_5(G) and L_5(T) are computed analogously

with $d_x = 0.1 \ \forall\ x \in \{1,..,5\}$, and $P_{ij}(0.1) = \begin{cases} 0.91 \text{ if } i = j \\ 0.03 \text{ for each } i \neq j \end{cases}$

Calculating tree likelihoods



	k
1	…C…
2	…G…
3	…C…
4	…C…

Node 1:

A	0
C	1
G	0
T	0

Node 2:

A	0
C	0
G	1
T	0

Node 3:

A	0
C	1
G	0
T	0

Node 4:

A	0
C	1
G	0
T	0

Node 5:

A	0.0009
C	0.0273
G	0.0273
T	0.0009

Node 6:

A	0.0000022
C	
G	
T	

For a single site pattern *k* and a given tree:

$$L_6(i) = \prod_{v=\{3,4,5\}} \left[\sum_{j=\{A,C,G,T\}} L_v(j) \times P_{ij}(d_v) \right], \forall\, i \in \{A,C,G,T\}$$ *

Likelihood of nucleotide **A** at node **6** in column **k**!

$$L_6(A) =$$

$$[L_5(A)xP_{AA}(d_5) + L_5(C)xP_{CA}(d_5) + L_5(G)xP_{GA}(d_5) + L_5(T)xP_{TA}(d_5)]x$$

$$[L_3(A)xP_{AA}(d_3) + L_3(C)xP_{CA}(d_3) + L_3(G)xP_{GA}(d_3) + L_3(T)xP_{TA}(d_3)]x$$

$$[L_4(A)xP_{AA}(d_4) + L_4(C)xP_{CA}(d_4) + L_4(G)xP_{GA}(d_4) + L_4(T)xP_{TA}(d_4)] =$$

$$[0.0009x0.91 + 0.0273x0.03 + 0.0273x0.03 + 0.0009x0.03]x$$

$$[0x0.91 + 1x0.03 + 0x0.03 + 0x0.03]x$$

$$[0x0.91 + 1x0.03 + 0x0.03 + 0x0.03] = 0.002484x0.03x0.03 =$$

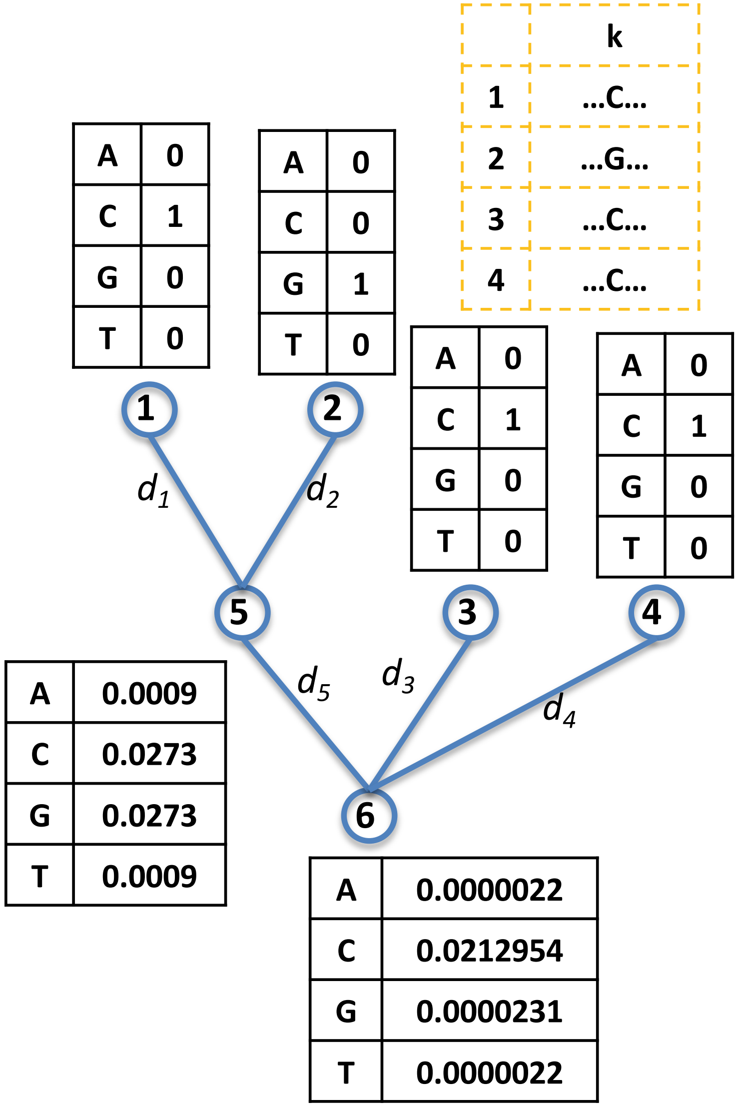
0.0000022

with $d_x = 0.1\ \forall\, x \in \{1,..,5\}$, and $P_{ij}(0.1) = \begin{cases} 0.91 \text{ if } i = j \\ 0.03 \text{ for each } i \neq j \end{cases}$

---

* Note, the v represents the nodes for which the partial likelihoods have already been computed. The sum indicates that you sum over all possible internal labels. Note, that for leaf nodes the probability of the observed nucleotide is 1 and that of the other nucleotides is 0! Hence, for nodes 3 and 4 there is no need to compute a sum!

Calculating tree likelihoods



	k
1	…C…
2	…G…
3	…C…
4	…C…

Node 1:

A	0
C	1
G	0
T	0

Node 2:

A	0
C	0
G	1
T	0

Node 3:

A	0
C	1
G	0
T	0

Node 4:

A	0
C	1
G	0
T	0

Node 5:

A	0.0009
C	0.0273
G	0.0273
T	0.0009

Node 6:

A	0.0000022
C	0.0212954
G	0.0000231
T	0.0000022

For a single site pattern *k* and a given tree:

$$L_5(i) = [L(C) \times P_{iC}(d_1)] \times [L(G) \times P_{iG}(d_2)], \forall\, i \in \{A,C,G,T\}$$

$$L_6(i) = \prod_{v=\{3,4,5\}} \left[\sum_{j=\{A,C,G,T\}} L_v(j) \times P_{ij}(d_v) \right], \forall\, i \in \{A,C,G,T\} \quad *$$

$$L^{(k)} = \sum_{i=\{A,C,G,T\}} \pi_i \times L_6(i) = 0.005331; \text{ mit } \pi_i = 0.25 \forall i \in \{A,G,C,T\}$$

This is the **site likelihood** of the pattern *k* given the tree

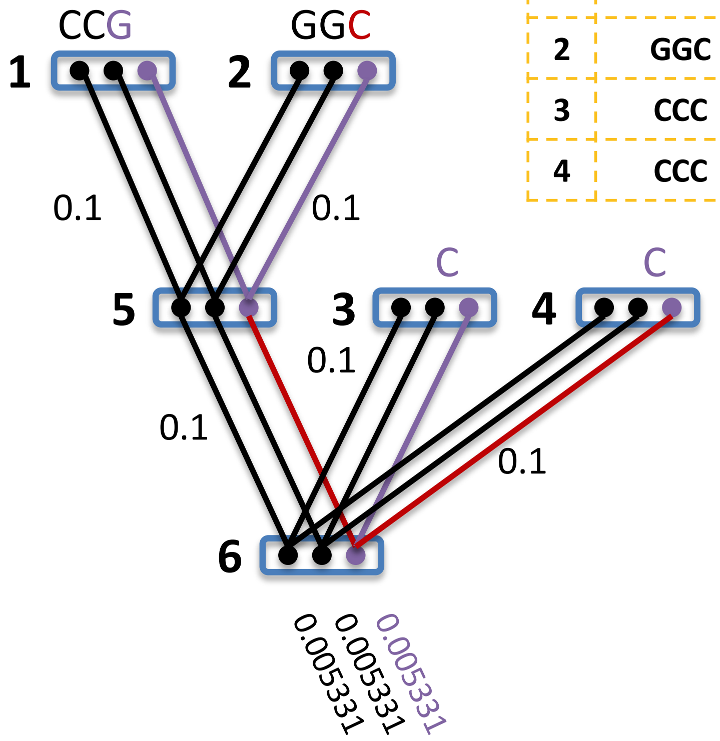
$$\text{with } d_x = 0.1 \;\forall\; x \in \{1,..,5\}, \text{ and } P_{ij}(0.1) = \begin{cases} 0.91 \text{ if } i = j \\ 0.03 \text{ for each } i \neq j \end{cases}$$

---

* Note, the v represents the nodes for which the partial likelihoods have already been computed. The sum indicates that you sum over all possible internal labels.

Calculating tree likelihoods

1	**CCG**
2	**GGC**
3	**CCC**
4	**CCC**



For an alignment of four sequences and length m=3 the likelihood is then

$$L(T)=\prod_{k=1}^{m} L^{(k)} = 0.005331^{2} \times 0.005331$$

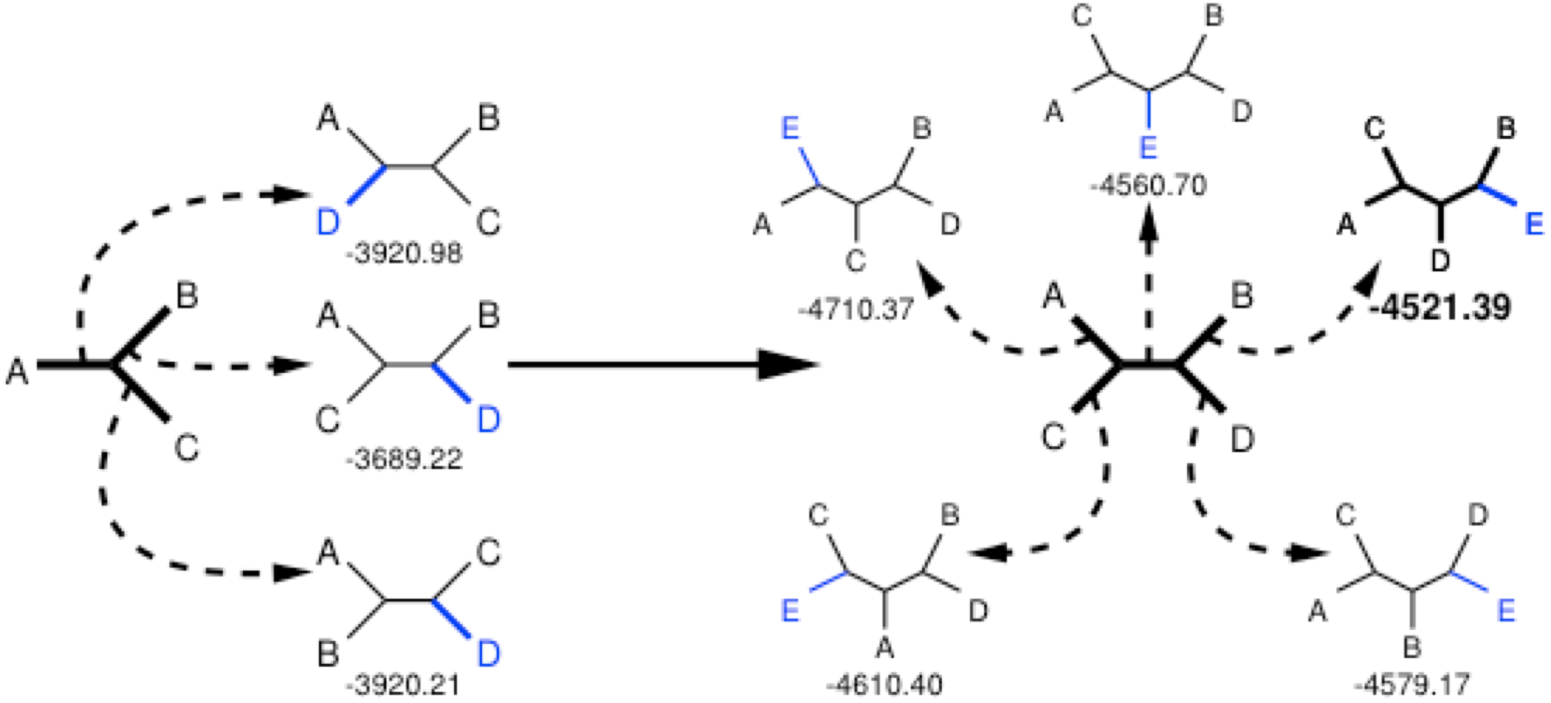
$$= 0.000000152$$

or the log-likelihood is

$$\ln L(T) = \sum_{k=1}^{m} \ln L^{(k)} = -15.7$$

Now that we know how to evaluate the likelihood of any given tree, we need to ask how to find the ML tree

Heuristic tree search begins with an initial sub-optimal solution (starting tree) obtained either via step-wise addition (or using a distance tree)

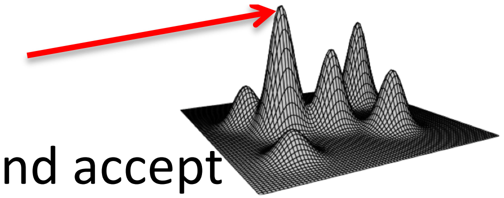


Maximum Parsimony and Maximum Likelihood only evaluate trees and do not reconstruct them!
Finding the best tree is highly complex!

1. **Exhaustive Search:** evaluates every possible tree and hence an optimal solution is guaranteed. Limit: 10-12 taxa

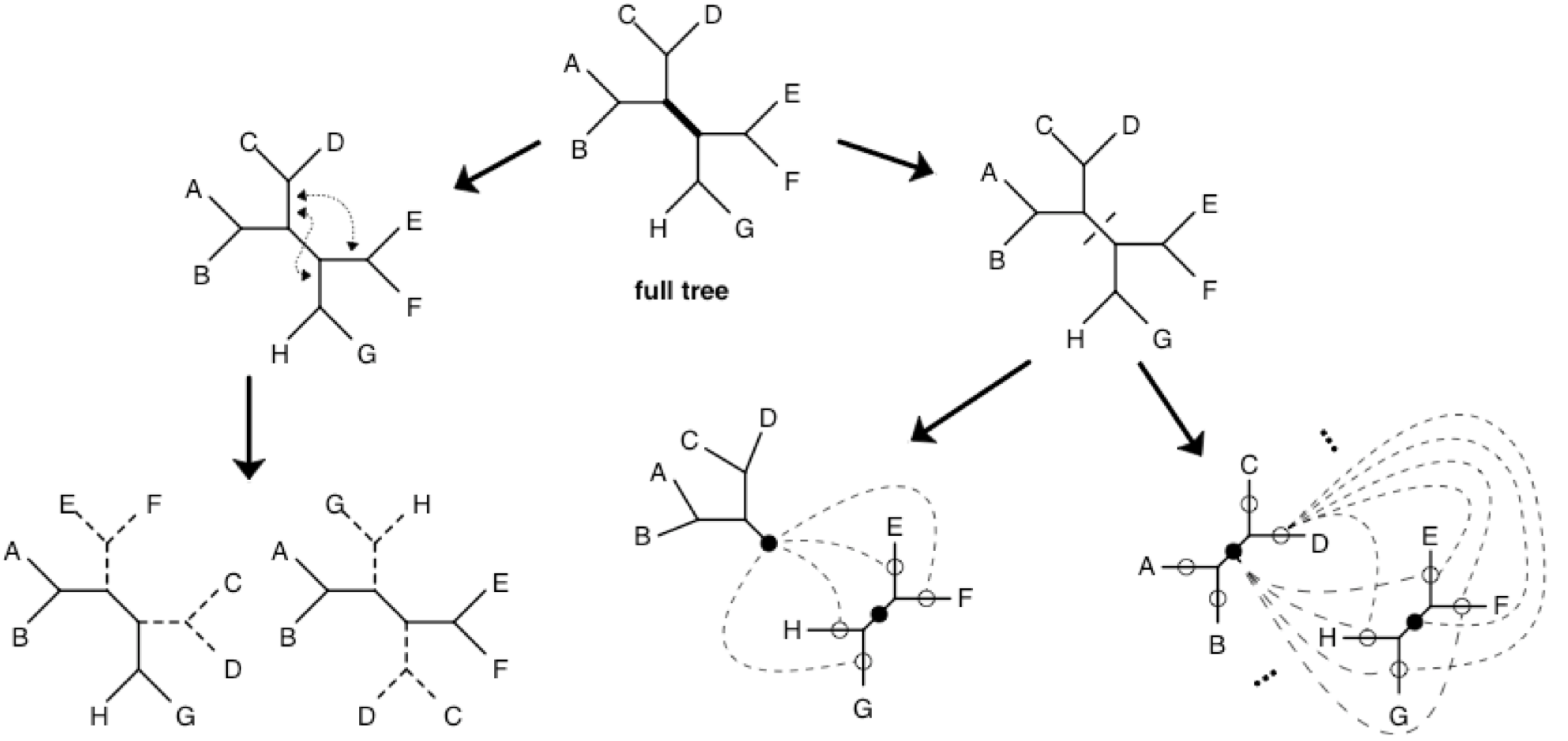
2. **Branch and Bound:** excludes parts from the tree space from the search where the optimal tree cannot be found. Guarantees to find the optimal tree.

3. **Heuristics:** Can be applied to large taxon sets but does not guarantee an optimal solution. Here, stochastic iterative algorithms are used that randomly modify a tree and accept the modification if a better tree is found.


our goal!

Finding the best tree

Evaluate random rearrangements of the starting tree and accept new tree if it improves P(D|M,T). Continue until convergence.



Nearest Neighbor Interchange (NNI)

O(n) NNI trees

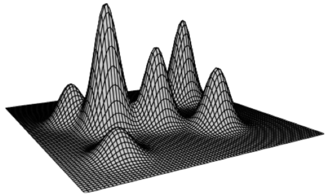
Subtree Pruning + Regrafting (SPR)

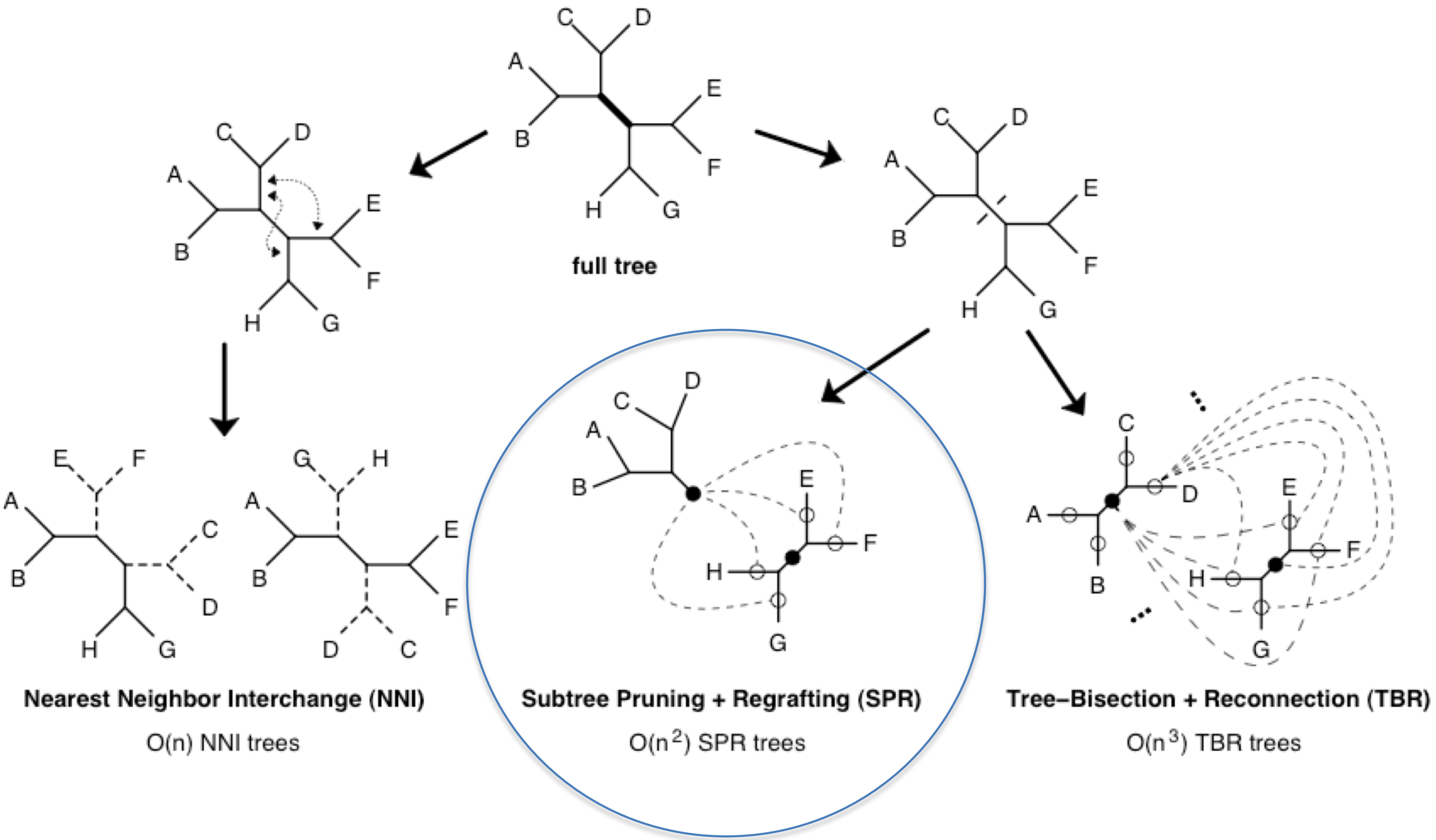
$O(n^2)$ SPR trees

Tree–Bisection + Reconnection (TBR)

$O(n^3)$ TBR trees

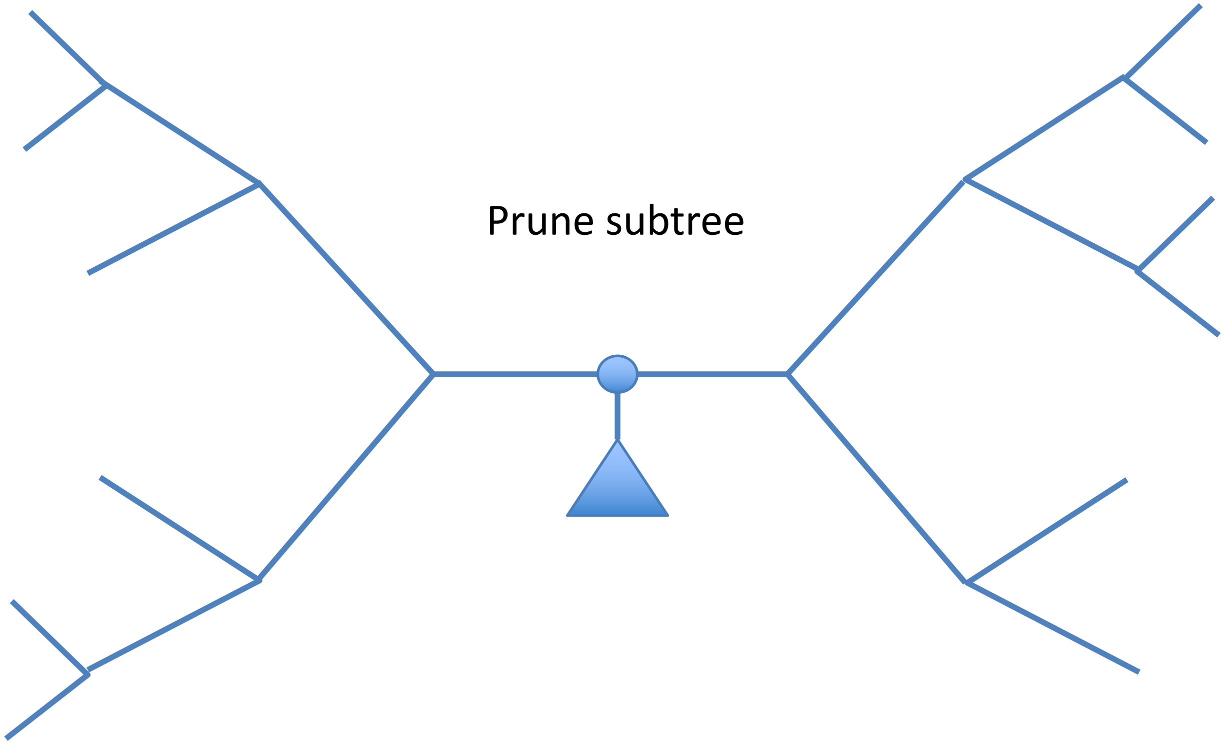
Tree rearrangements in RAxML*





Stamatakis et al. 2008 Syst Biol 57:758-771

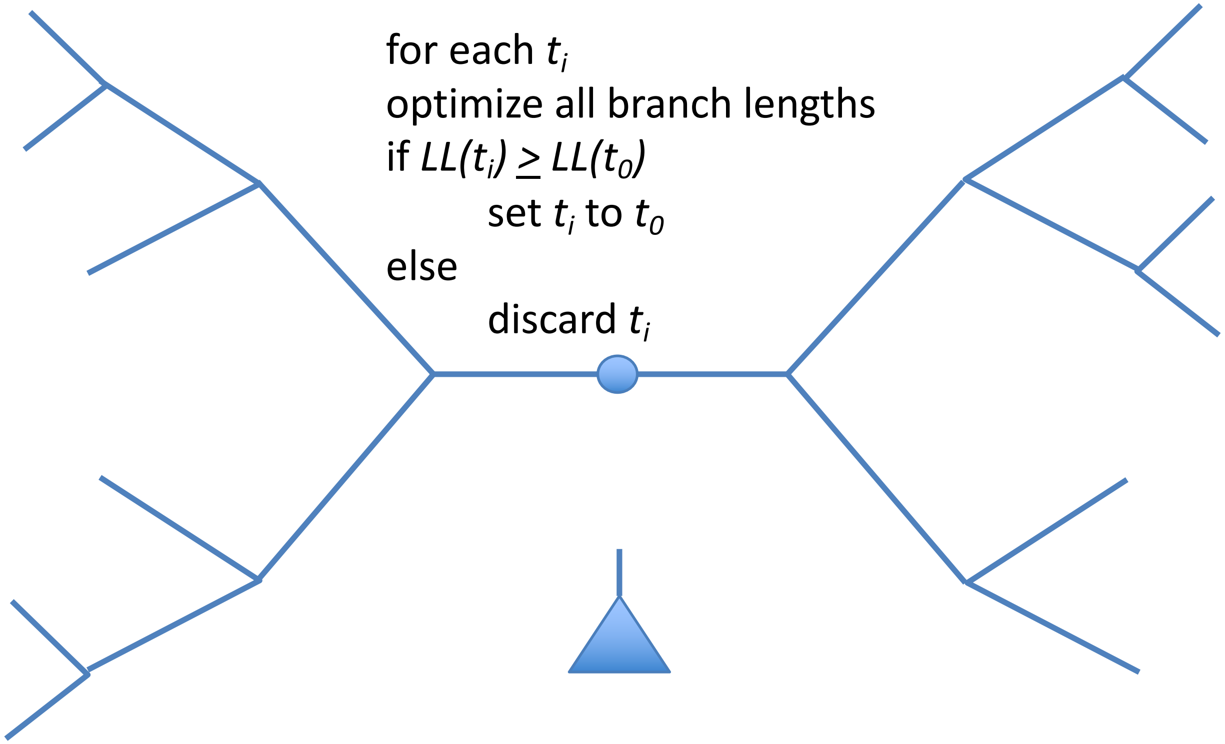
Lazy subtree rearrangement in RaxML


Prune subtree

Stamatakis et al. 2008 Syst Biol 57:758-771

Subtree pruning and regrafting
(a single iteration)

Regraft subtree subsequently on all branches to form all alternative topologies t_i



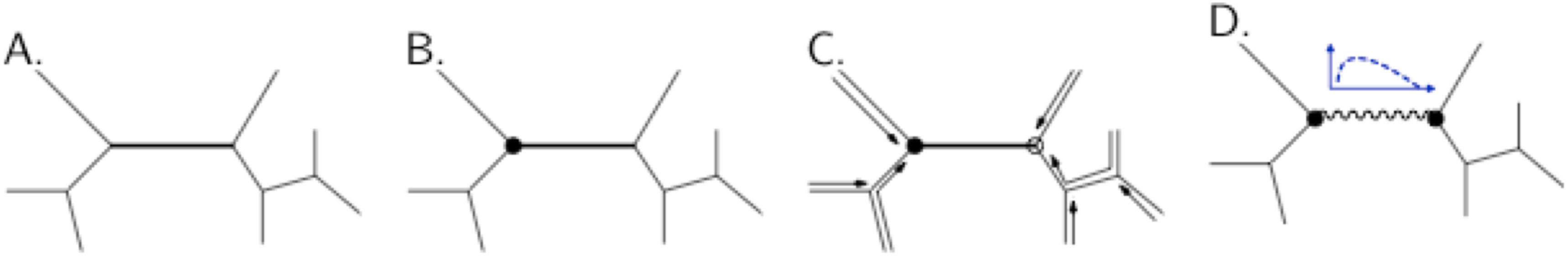
```
for each $t_i$
optimize all branch lengths
if $LL(t_i) \geq LL(t_0)$
        set $t_i$ to $t_0$
else
        discard $t_i$
```

Stamatakis et al. 2008 Syst Biol 57:758-771; *LL represents LogLikelihood*

Optimizing branch lengths is time consuming due to re-calculation of all the partial likelihoods

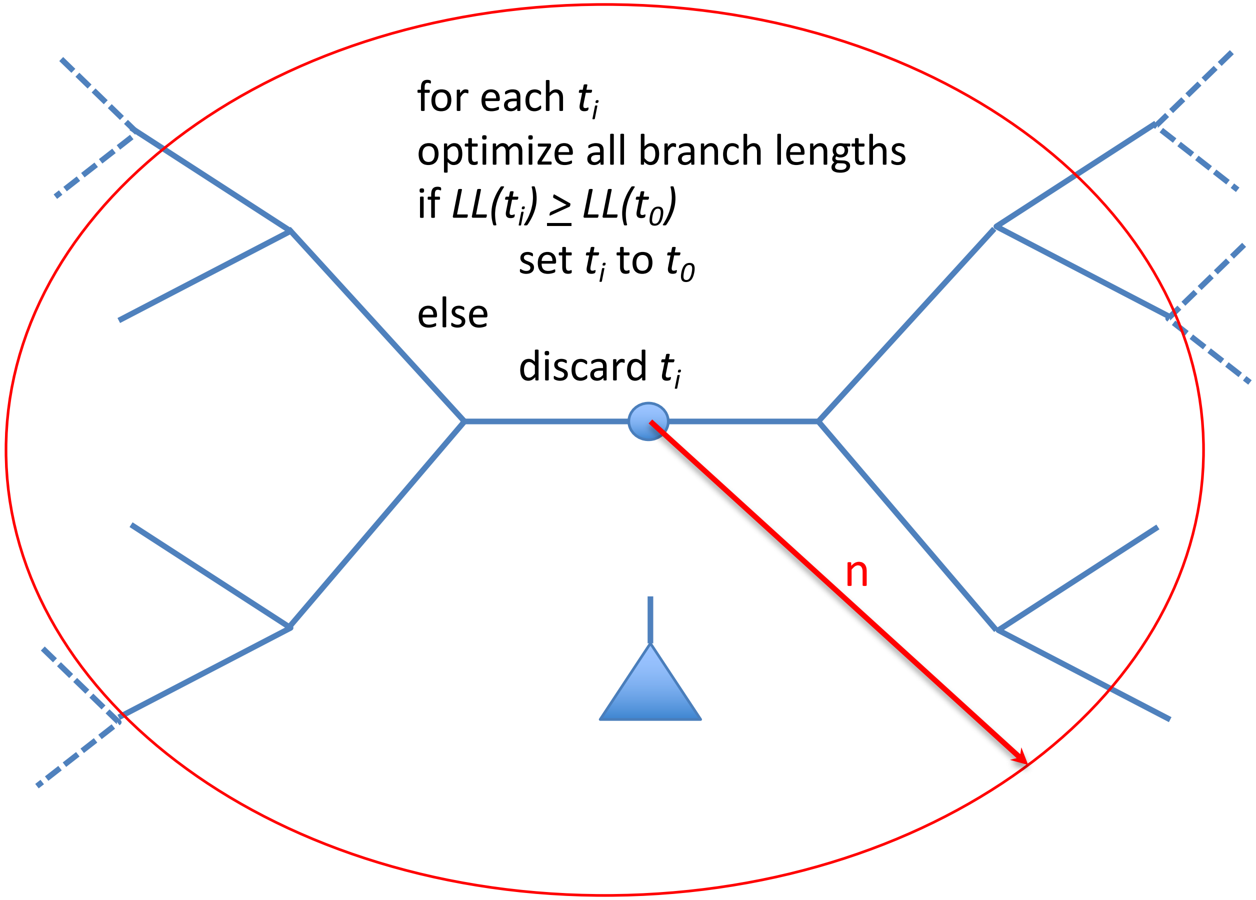
To compute optimal branch lengths do the following after initializing all branch lengths.

A. Choose a branch
B. Move the virtual root to an adjacent node
C. Compute all partial likelihoods recursively
D. Adjust the branch length to maximize the likelihood value



Subtree Rearrangement in RAxML
(a single iteration)

Regraft subtree subsequently only on branches up to a certain distance *n** to form t_i



```
for each $t_i$
optimize all branch lengths
if $LL(t_i) \geq LL(t_0)$
        set $t_i$ to $t_0$
else
        discard $t_i$
```

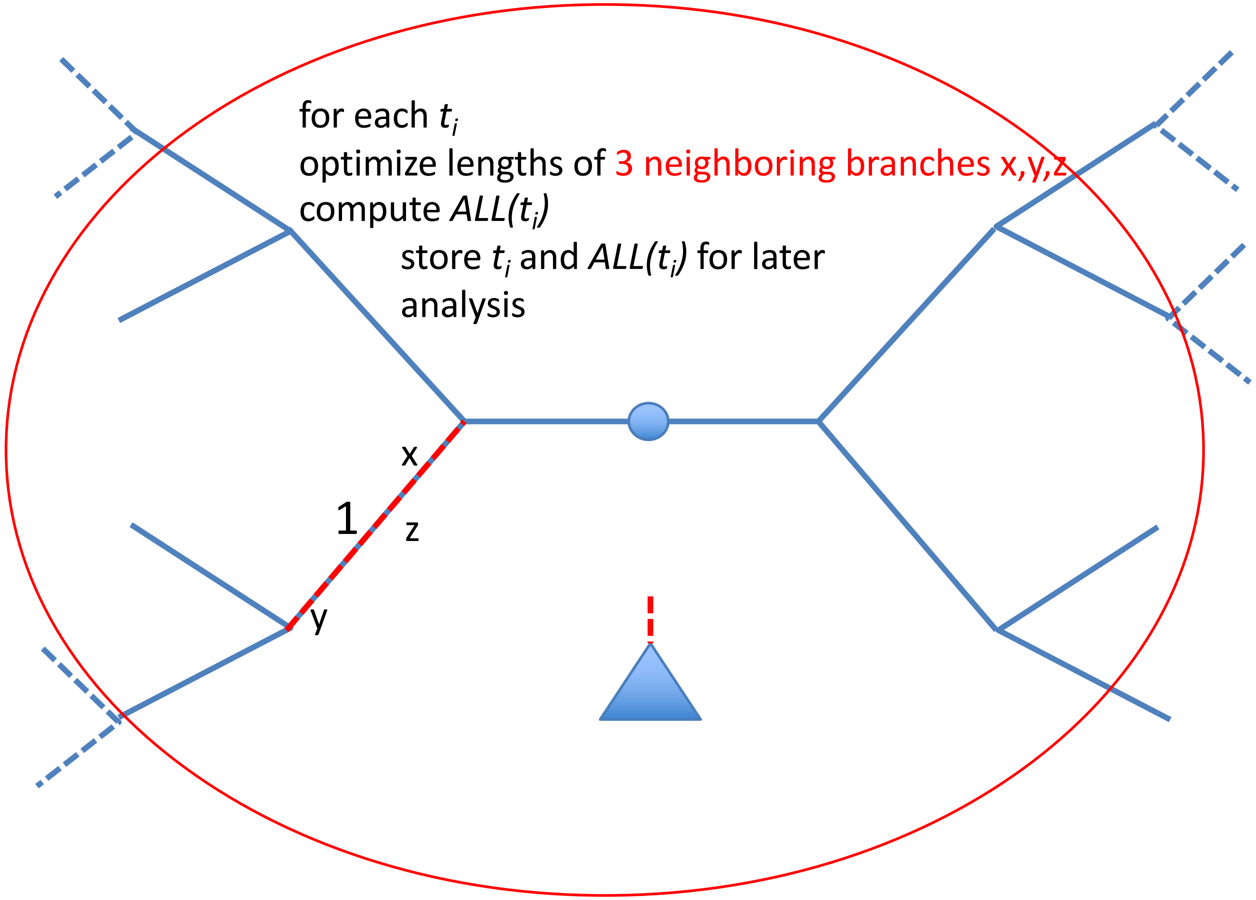
Rationale: Reduce the search space for better trees (computational speed-up)

---

*radius of n internal nodes from pruning point. *n* takes typically values between 5 and 25

Lazy Subtree Rearrangement in RAxML
(a single iteration)

Regraft subtree subsequently on all branches up to a certain distance *n* to form t_i

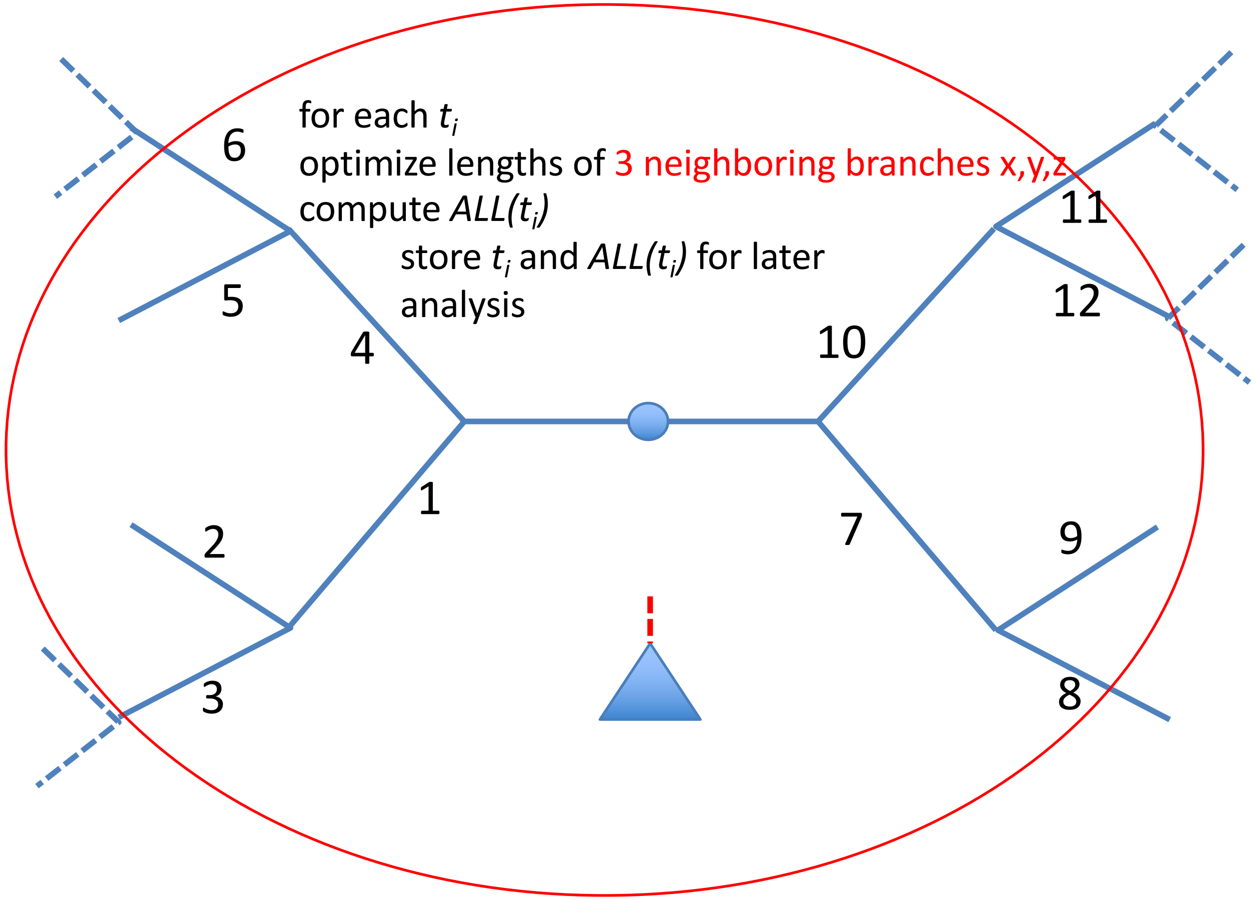

for each t_i
optimize lengths of 3 neighboring branches x,y,z
compute $ALL(t_i)$
store t_i and $ALL(t_i)$ for later analysis

Tree	ALL
t_1	$ALL(t_1)$

Rationale: Reduce computational burden by computing only **A**pproximate **L**og **L**ikelihoods (ALL) to rapidly pre-screen for promising topologies.

Lazy Subtree Rearrangement in RAxML

Regraft subtree subsequently on all branches up to a certain distance *n* to form t_i



for each t_i
optimize lengths of 3 neighboring branches x,y,z
compute $ALL(t_i)$
    store t_i and $ALL(t_i)$ for later analysis

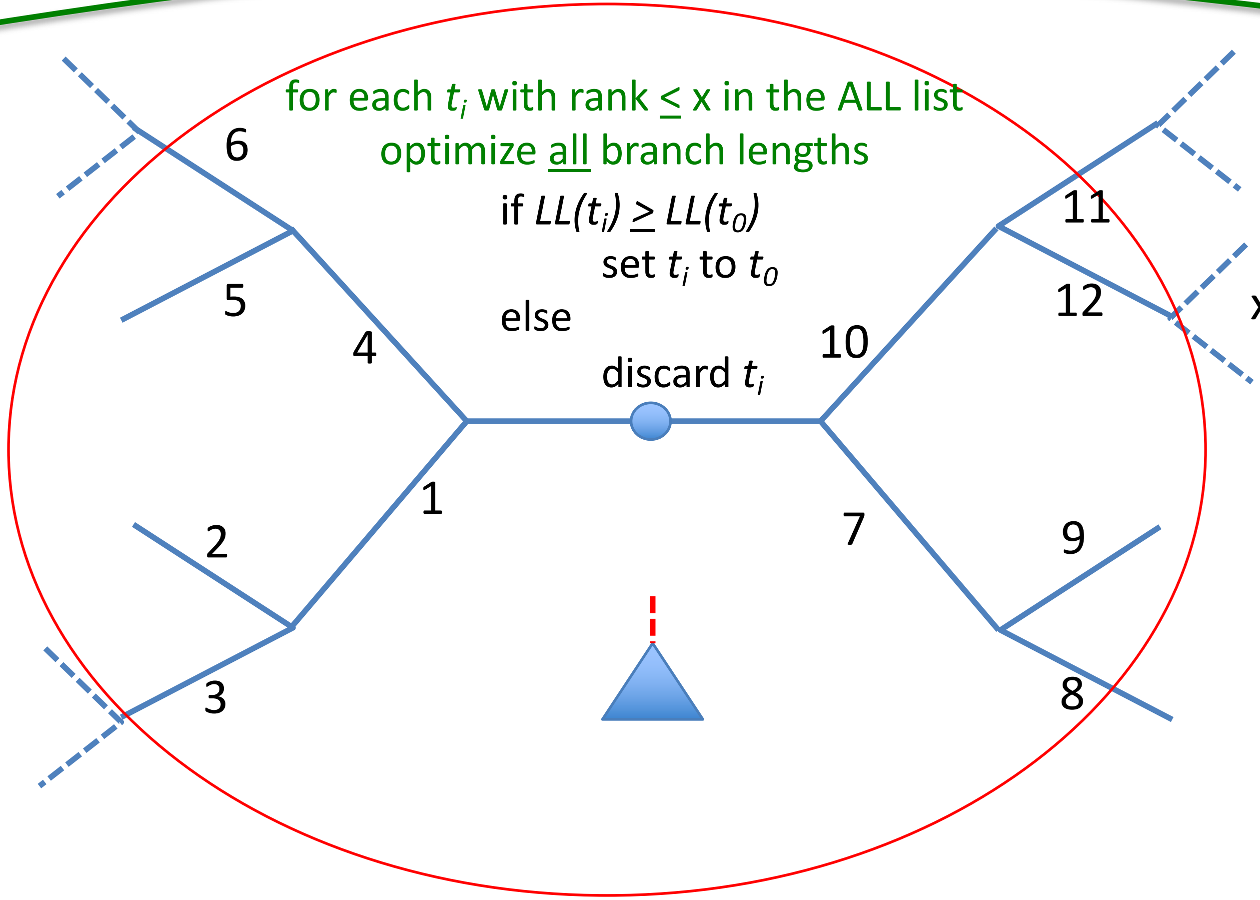
Tree	ALL
t_1	$ALL(t_1)$
t_2	$ALL(t_2)$
t_3	$ALL(t_3)$
t_4	$ALL(t_4)$
t_5	$ALL(t_5)$
t_6	$ALL(t_6)$
t_7	$ALL(t_7)$
t_8	$ALL(t_8)$
t_9	$ALL(t_9)$
t_{10}	$ALL(t_{10})$
t_{11}	$ALL(t_{11})$
t_{12}	$ALL(t_{12})$

Rationale: Reduce computational burden by computing only **A**pproximate **L**og **L**ikelihoods (ALL) to rapidly pre-screen for promising topologies.

Lazy Subtree Rearrangement in RAxML

Sort list of t_i according to $ALL(t_i)$ and choose the x best trees for thorough optimization

Tree	LL
t_1	LL(t_1)
t_2	LL(t_2)
t_9	LL(t_9)
t_7	LL(t_7)
t_5	LL(t_5)
t_6	LL(t_6)
t_4	LL(t_4)



```
for each $t_i$ with rank $\leq$ x in the ALL list
    optimize all branch lengths
    if $LL(t_i) \geq LL(t_0)$
        set $t_i$ to $t_0$
    else
        discard $t_i$
```

Tree	ALL
t_1	ALL(t_1)
t_2	ALL(t_2)
t_9	ALL(t_9)
t_7	ALL(t_7)
t_5	ALL(t_5)
t_6	ALL(t_6)
t_4	ALL(t_4)
t_8	ALL(t_8)
t_3	ALL(t_3)
t_{10}	ALL(t_{10})
t_{11}	ALL(t_{11})
t_{12}	ALL(t_{12})

x*

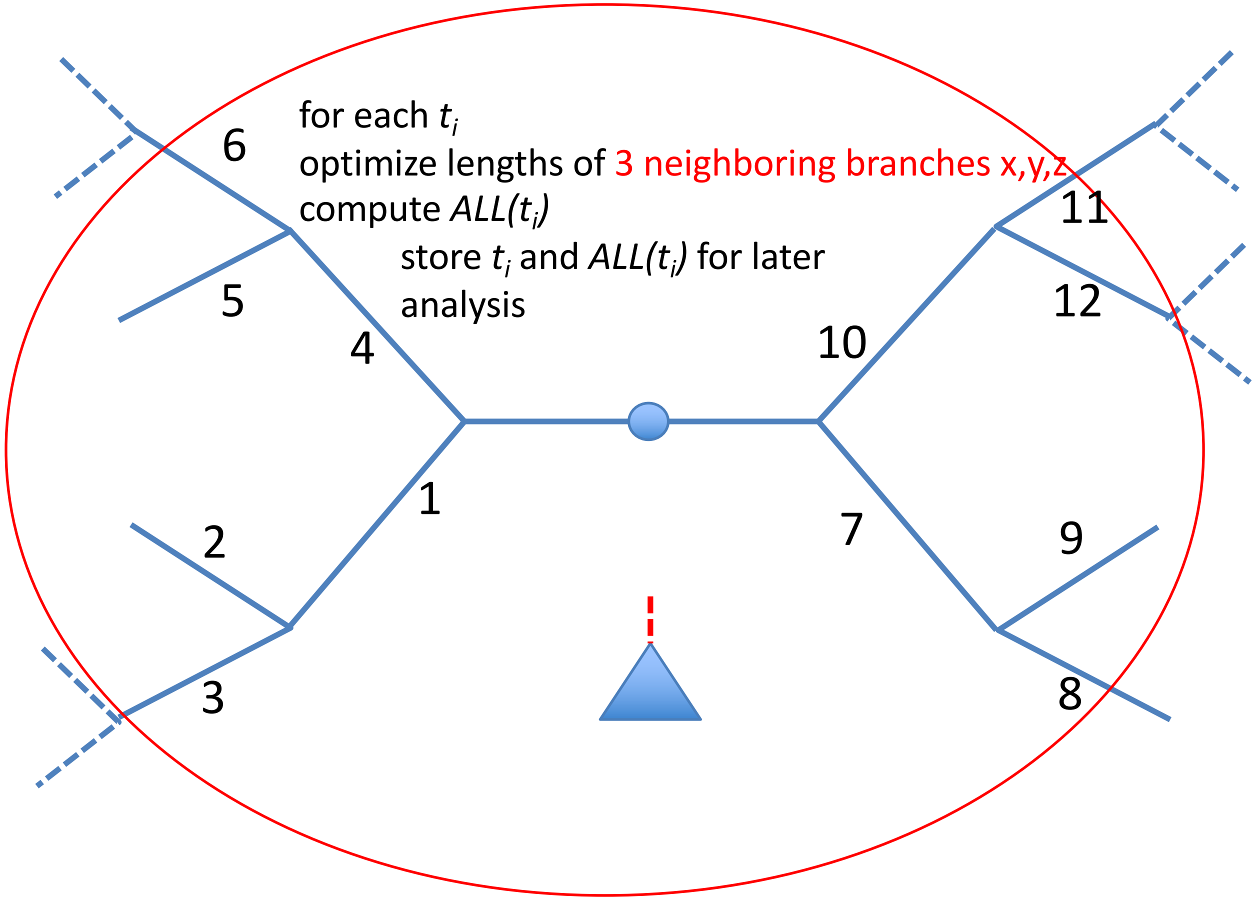
desc. ALL

Rationale: Reduce computational burden by computing only **Approximate Log Likelihoods** (ALL) to rapidly pre-screen for promising topologies.

*x typically set to 20

Lazy Subtree Rearrangement in RAxML
Likelihood Cutoff Heuristics

Regraft subtree subsequently on all branches up to a certain distance *n* to form t_i



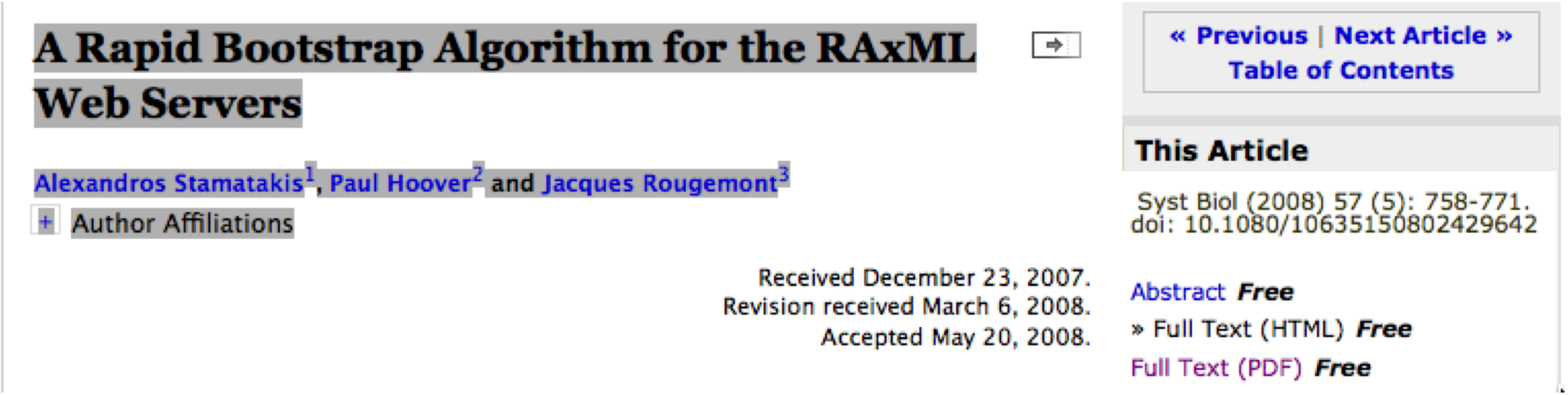
for each t_i

optimize lengths of 3 neighboring branches x,y,z

compute $ALL(t_i)$

store t_i and $ALL(t_i)$ for later analysis

Tree	ALL
t_1	ALL(t_1)
t_2	ALL(t_2)
t_3	ALL(t_3)
t_4	ALL(t_4)
t_5	ALL(t_5)
t_6	ALL(t_6)
t_7	ALL(t_7)
t_8	ALL(t_8)
t_9	ALL(t_9)
t_{10}	
t_{11}	
t_{12}	



A Rapid Bootstrap Algorithm for the RAxML Web Servers

Alexandros Stamatakis[1], Paul Hoover[2] and Jacques Rougemont[3]

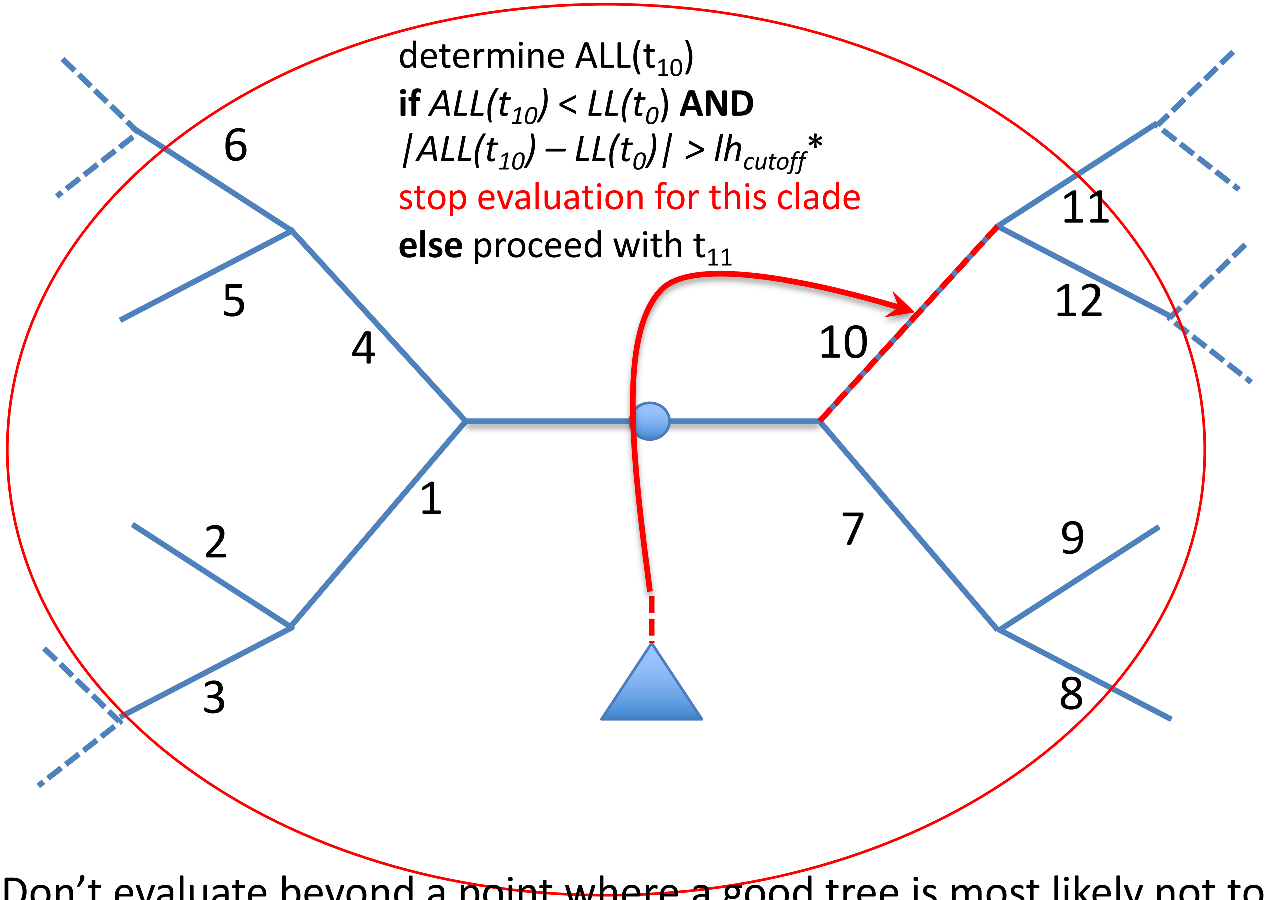
Received December 23, 2007.
Revision received March 6, 2008.
Accepted May 20, 2008.

Syst Biol (2008) 57 (5): 758-771.
doi: 10.1080/10635150802429642

Thus, if the approximate log likelihood all (t′) for the rearranged tree t′ is worse than the log likelihood ll (t) of the currently best tree t and if the difference δ(all (t′), ll(t)), where δ(x, y)=x−y, is larger than a certain—dynamically determined—threshold lhcutoff, the remaining LSRs beyond that node are omitted.

Lazy Subtree Rearrangement in RAxML
Likelihood Cutoff Heuristics

Regraft subtree subsequently on all branches up to a certain distance *n* to form t_i



determine ALL(t_{10})
if $ALL(t_{10}) < LL(t_0)$ **AND**
$|ALL(t_{10}) - LL(t_0)| > lh_{cutoff}$*
stop evaluation for this clade
else proceed with t_{11}

Tree	ALL
t_1	ALL(t_1)
t_2	ALL(t_2)
t_3	ALL(t_3)
t_4	ALL(t_4)
t_5	ALL(t_5)
t_6	ALL(t_6)
t_7	ALL(t_7)
t_8	ALL(t_8)
t_9	ALL(t_9)
t_{10}	ALL(t_{10})
t_{11}	-
t_{12}	-

Rationale: Don't evaluate beyond a point where a good tree is most likely not to be found. In other words, if t_{10} is already bad there is no need to evaluate t_{11} or t_{12} belonging to the same clade.

---

*lh_{cutoff} is dynamically determined

Lazy Subtree Rearrangement in RAxML
Likelihood Cutoff Heuristics

Determining dynamically the likelihood cutoff value lh_{cutoff}

Iteration 1:

1. initialize $\mathrm{lh}_{\mathrm{cutoff}}$ with ∞
2. perform a full descent into all alternative topologies within rearrangement distance *n*
3. for each alternative tree topology t_i compute $d_i=|ALL(t_i) – LL(t_0)|$
4. compute lh_{cutoff} as

$$lh_{cutoff} = \sum_{i=1}^{m} d_i / m$$

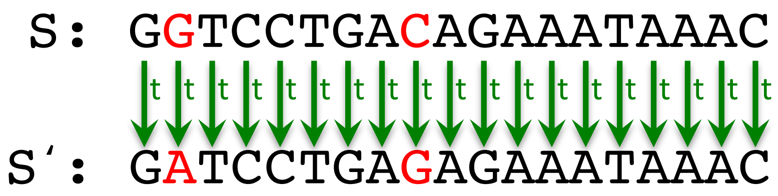
where *m* is the number of evaluated alternative topologies

Iteration k > 1:

1. set lh_{cutoff} to the value determined in iteration *k-1*
2. for all clades
   1. evaluate alternative tree topologies t_i at increasing rearrangement distance from t_0
   2. compute $d_i=|ALL(t_i) – LL(t_0)|$ and stop descent into clade when $d_i > lh_{cutoff}$
3. update lh_{cutoff}

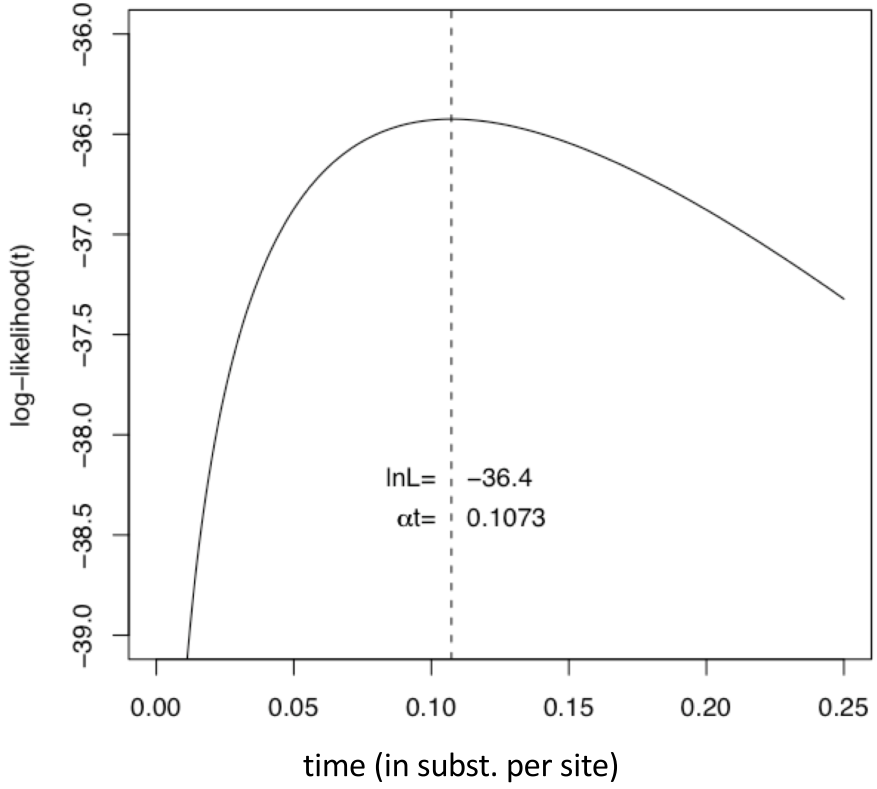
How stable is my tree?

Remember: Our assumption, thus far, was that all columns in our alignment share the same evolutionary history (here denoted by the same time *t* for each site)



Multiply 'site-likelihoods' across all *m* positions of the alignment! The underlying assumption is: **All positions evolved according to the same tree!**

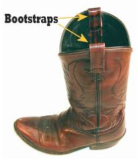
$$L(t \mid s \rightarrow s') = \boxed{\prod_{i=1}^{m}} \left(\pi_{s_i} \times P_{s_i s_i'}(t)\right)$$


Log-Likelihood surface under JC69

---

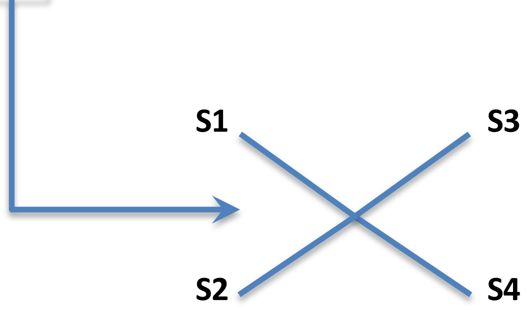
*watch out, products of probabilities become small very quckly. We therefore compute typically in log scale, hence the log-likelihoods

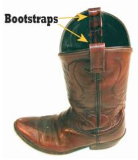
Resampling methods for assessing the support of a tree given the data



Rationale: All positions in a sequence, and hence all alignment columns, should have the same evolutionary history. Thus, it should in principle not matter which subset of the data I am using for tree reconstruction if the phylogenetic signal is sufficiently strong and indeed consistent.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

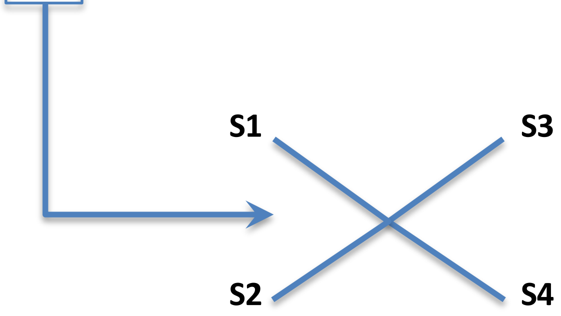




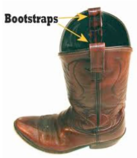
Resampling methods for assessing the support of a tree given the data

Rationale: All positions in a sequence, and hence all alignment columns, should have the same evolutionary history. Thus, it should in principle not matter which subset of the data I am using for tree reconstruction if the phylogenetic signal is sufficiently strong and indeed consistent.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

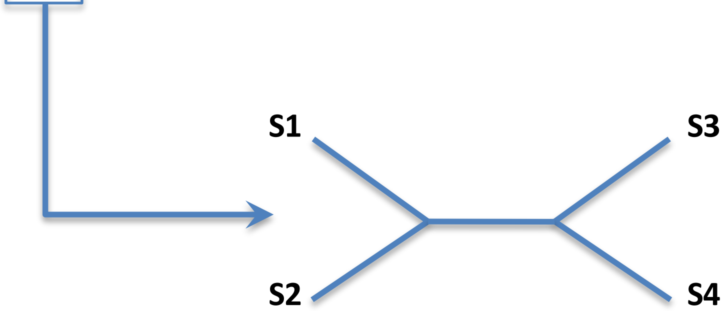


Resampling methods for assessing the support of a tree given the data

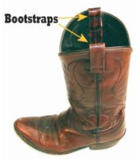


Rationale: All positions in a sequence, and hence all alignment columns, should have the same evolutionary history. Thus, it should in principle not matter which subset of the data I am using for tree reconstruction if the phylogenetic signal is sufficiently strong <u>and</u> indeed consistent.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

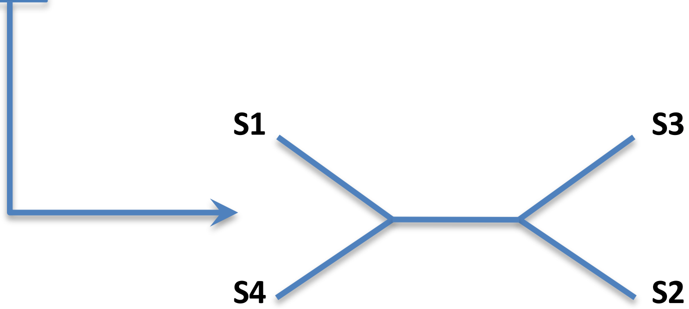


Resampling methods for assessing the support of a tree given the data

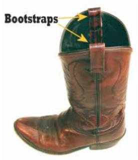


Rationale: All positions in a sequence, and hence all alignment columns, should have the same evolutionary history. Thus, it should in principle not matter which subset of the data I am using for tree reconstruction if the phylogenetic signal is sufficiently strong <u>and</u> indeed consistent.

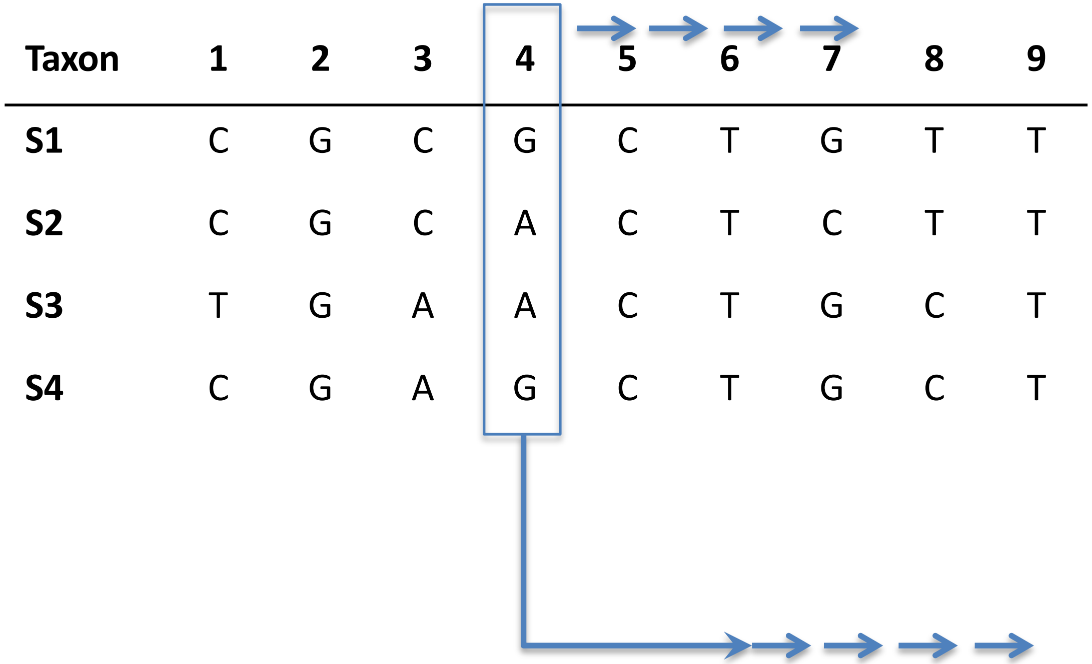
Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T



Resampling methods for assessing the support of a tree given the data

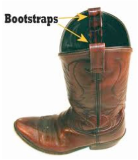


Rationale: All positions in a sequence, and hence all alignment columns, should have the same evolutionary history. Thus, it should in principle not matter which subset of the data I am using for tree reconstruction if the phylogenetic signal is sufficiently strong and indeed consistent.



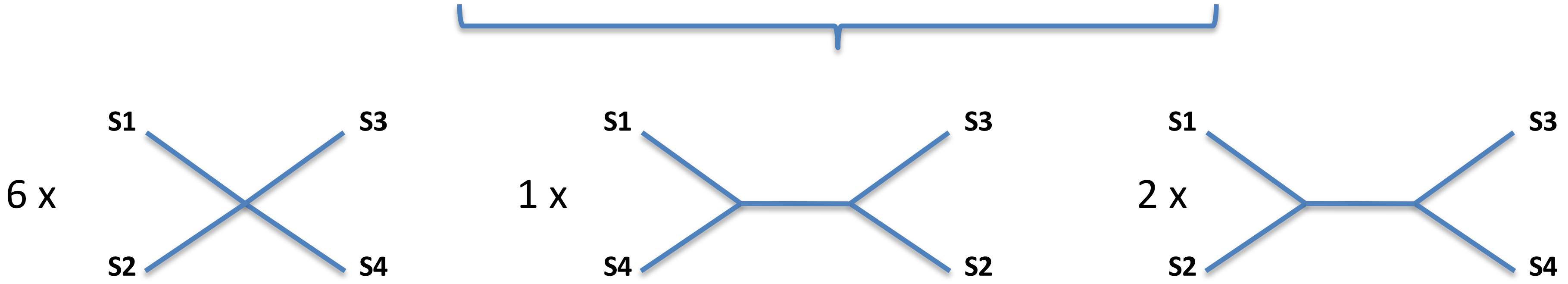
Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

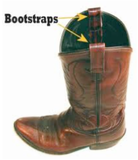
Resampling methods for assessing the support of a tree given the data



Observation: The phylogenetic signal in the data is apparently not entirely consistent and we would like to have a method to assess the extent of variability.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

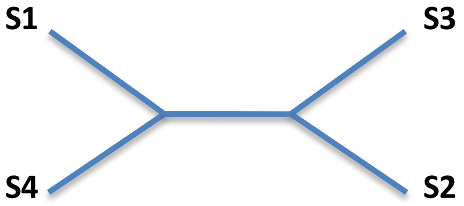




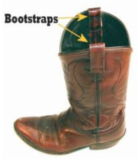
Resampling methods for assessing the support of a tree given the data

Approach 1 – Jackknife: Remove a random subset of alignment columns and re-compute the tree. Typically a 50% Jackknife analysis is performed.

Taxon	1	2	3	4	5	6	7	8	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

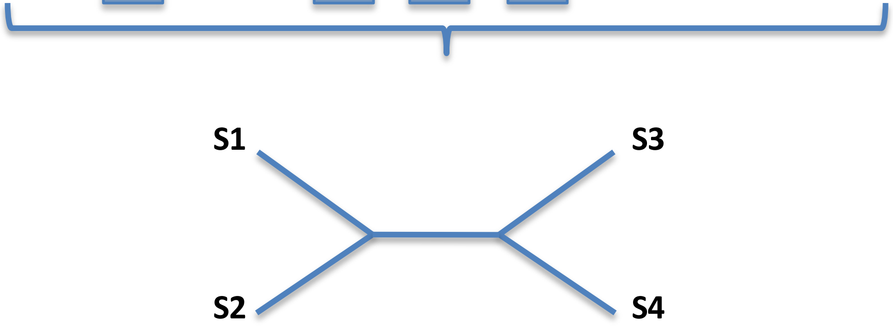


Resampling methods for assessing the support of a tree given the data

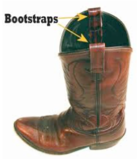


Approach 1 – Jackknife: Remove a random subset of alignment columns and re-compute the tree. Typically a 50% Jackknife analysis is performed.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

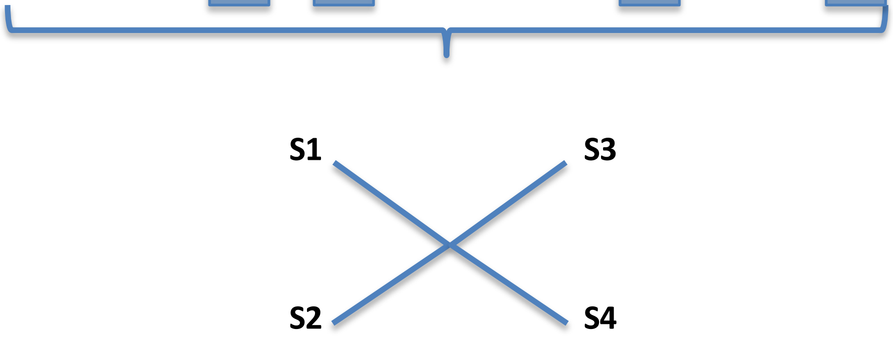


Resampling methods for assessing the support of a tree given the data

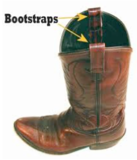


Approach 1 – Jackknife: Remove a random subset of alignment columns and re-compute the tree. Typically a 50% Jackknife analysis is performed.

Taxon	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T



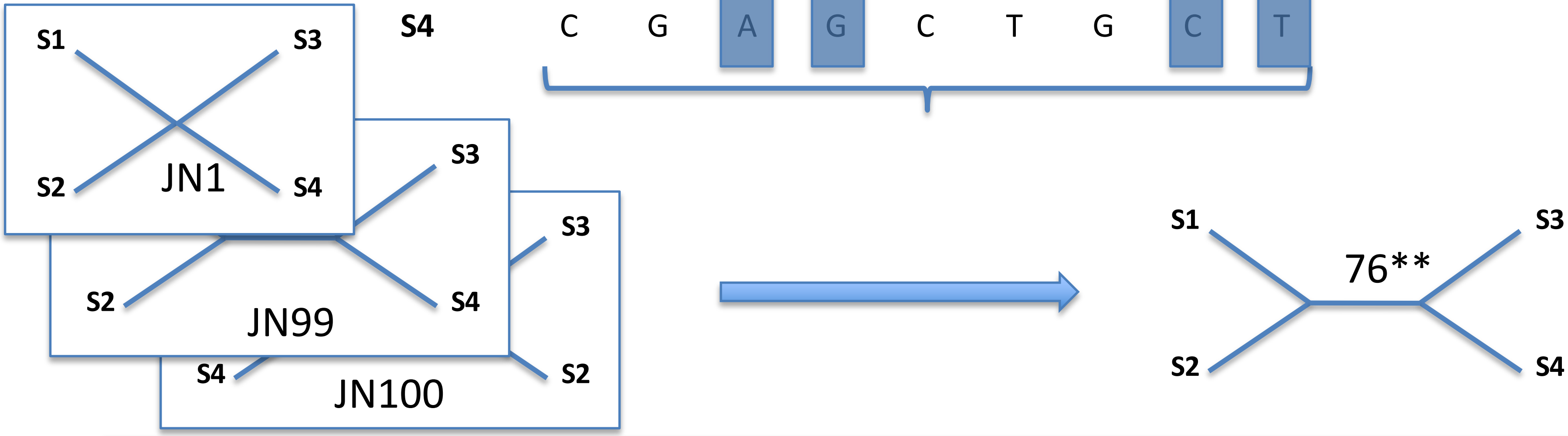
Resampling methods for assessing the support of a tree given the data



Approach 1 – Jackknife: Remove a random subset of alignment columns and re-compute the tree. Typically a 50% Jackknife analysis is performed.

repeat n* times

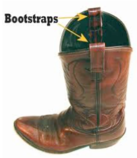
Taxon	1	2	3	4	5	6	7	8	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T



*n is typically 100 or 1000

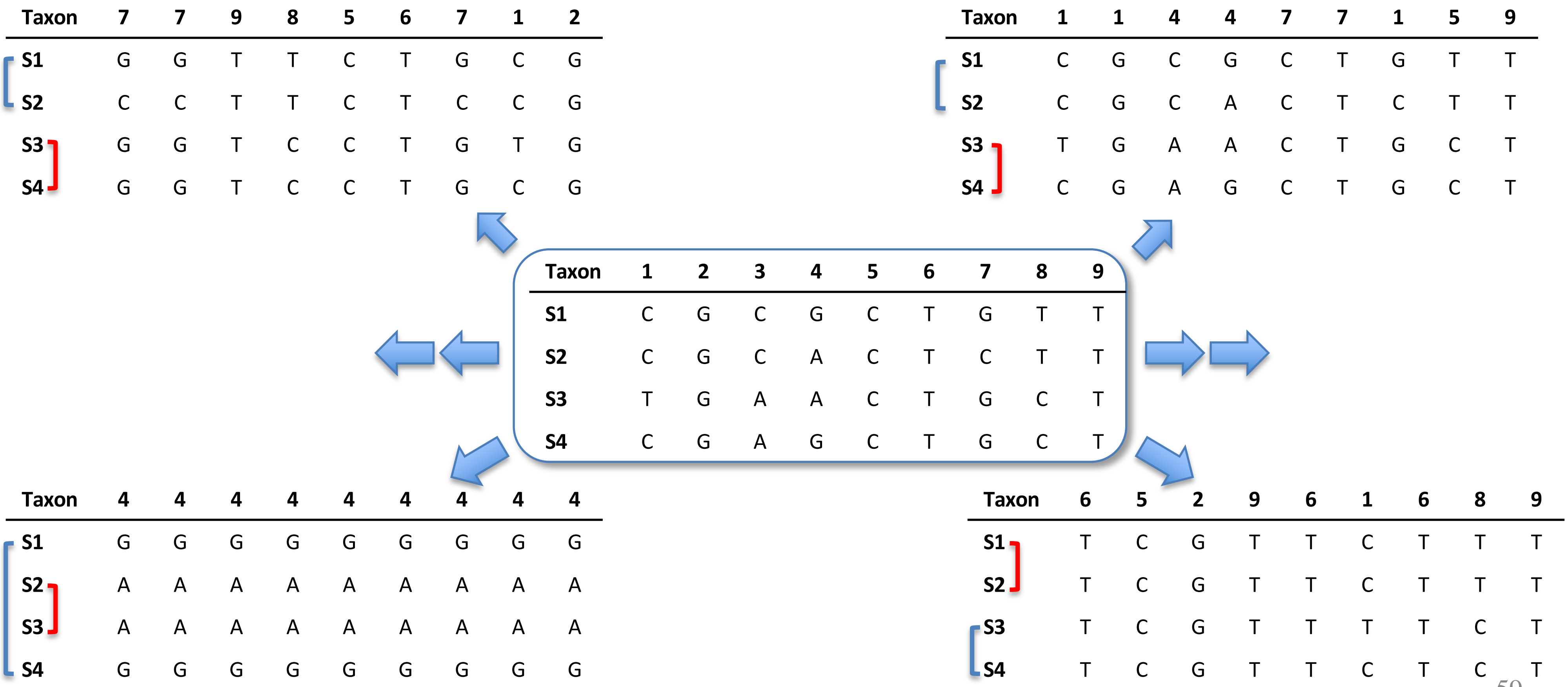
**value is typically given in percent

Resampling methods for assessing the support of a tree given the data



Approach 2 – Bootstrap: Resample randomly chosen columns from the original alignment (with replacement) to obtain a new alignment with the <u>same length</u> as the original alignment.

repeat n* times



Taxon	7	7	9	8	5	6	7	1	2
S1	G	G	T	T	C	T	G	C	G
S2	C	C	T	T	C	T	C	C	G
S3	G	G	T	C	C	T	G	T	G
S4	G	G	T	C	C	T	G	C	G

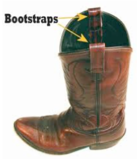
Taxon	1	1	4	4	7	7	1	5	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

Taxon	1	2	3	4	5	6	7	8	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

Taxon	4	4	4	4	4	4	4	4	4
S1	G	G	G	G	G	G	G	G	G
S2	A	A	A	A	A	A	A	A	A
S3	A	A	A	A	A	A	A	A	A
S4	G	G	G	G	G	G	G	G	G

Taxon	6	5	2	9	6	1	6	8	9
S1	T	C	G	T	T	C	T	T	T
S2	T	C	G	T	T	C	T	T	T
S3	T	C	G	T	T	T	T	C	T
S4	T	C	G	T	T	C	T	C	T

*n is typically 100 or 1000



Resampling methods for assessing the support of a tree given the data

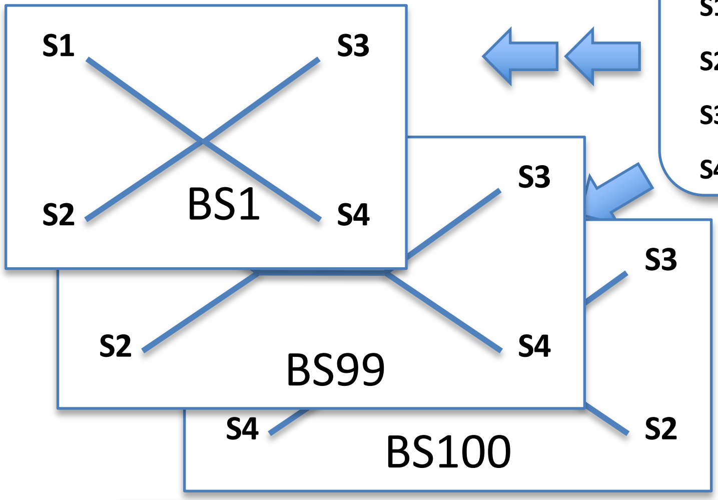
Approach 2 – Bootstrap: Resample randomly chosen columns from the original alignment (with replacement) to obtain a new alignment with the same length as the original alignment.

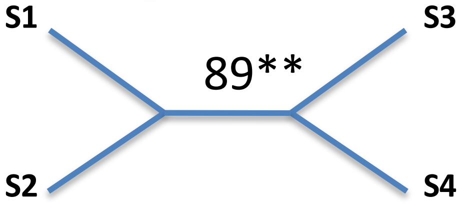
repeat n* times

Taxon	7	7	9	8	5	6	7	1	2
S1	G	G	T	T	C	T	G	C	G
S2	C	C	T	T	C	T	C	C	G
S3	G	G	T	C	C	T	G	T	G
S4	G	G	T	C	C	T	G	C	G

Taxon	1	1	4	4	7	7	1	5	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

Taxon	1	2	3	4	5	6	7	8	9
S1	C	G	C	G	C	T	G	T	T
S2	C	G	C	A	C	T	C	T	T
S3	T	G	A	A	C	T	G	C	T
S4	C	G	A	G	C	T	G	C	T

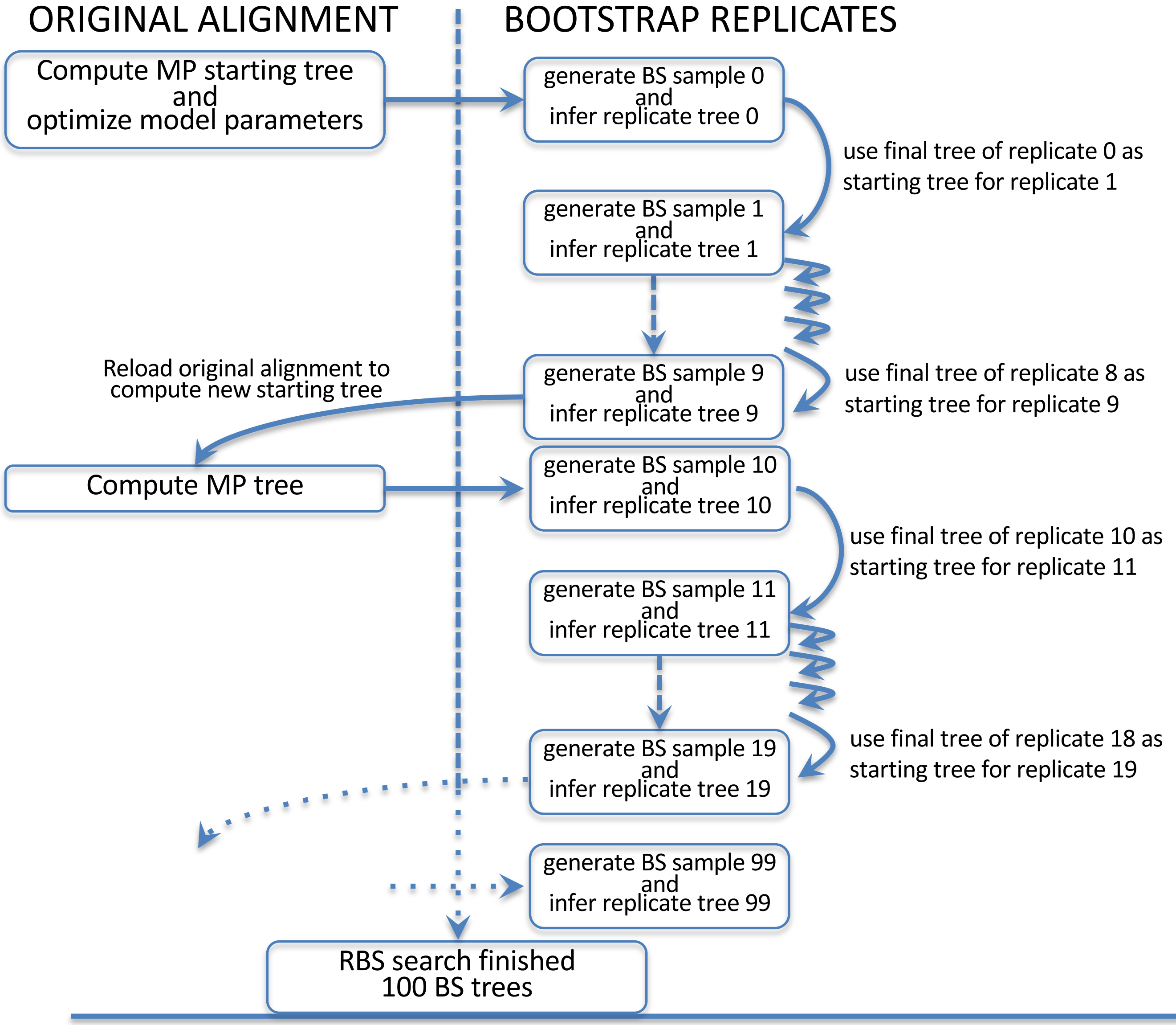




*n is typically 100 or 1000

**value is typically given in percent

Rapid bootstrapping in RAxML



ORIGINAL ALIGNMENT

BOOTSTRAP REPLICATES

Compute MP starting tree and optimize model parameters

generate BS sample 0 and infer replicate tree 0

use final tree of replicate 0 as starting tree for replicate 1

generate BS sample 1 and infer replicate tree 1

Reload original alignment to compute new starting tree

generate BS sample 9 and infer replicate tree 9

use final tree of replicate 8 as starting tree for replicate 9

Compute MP tree

generate BS sample 10 and infer replicate tree 10

use final tree of replicate 10 as starting tree for replicate 11

generate BS sample 11 and infer replicate tree 11

generate BS sample 19 and infer replicate tree 19

use final tree of replicate 18 as starting tree for replicate 19

generate BS sample 99 and infer replicate tree 99

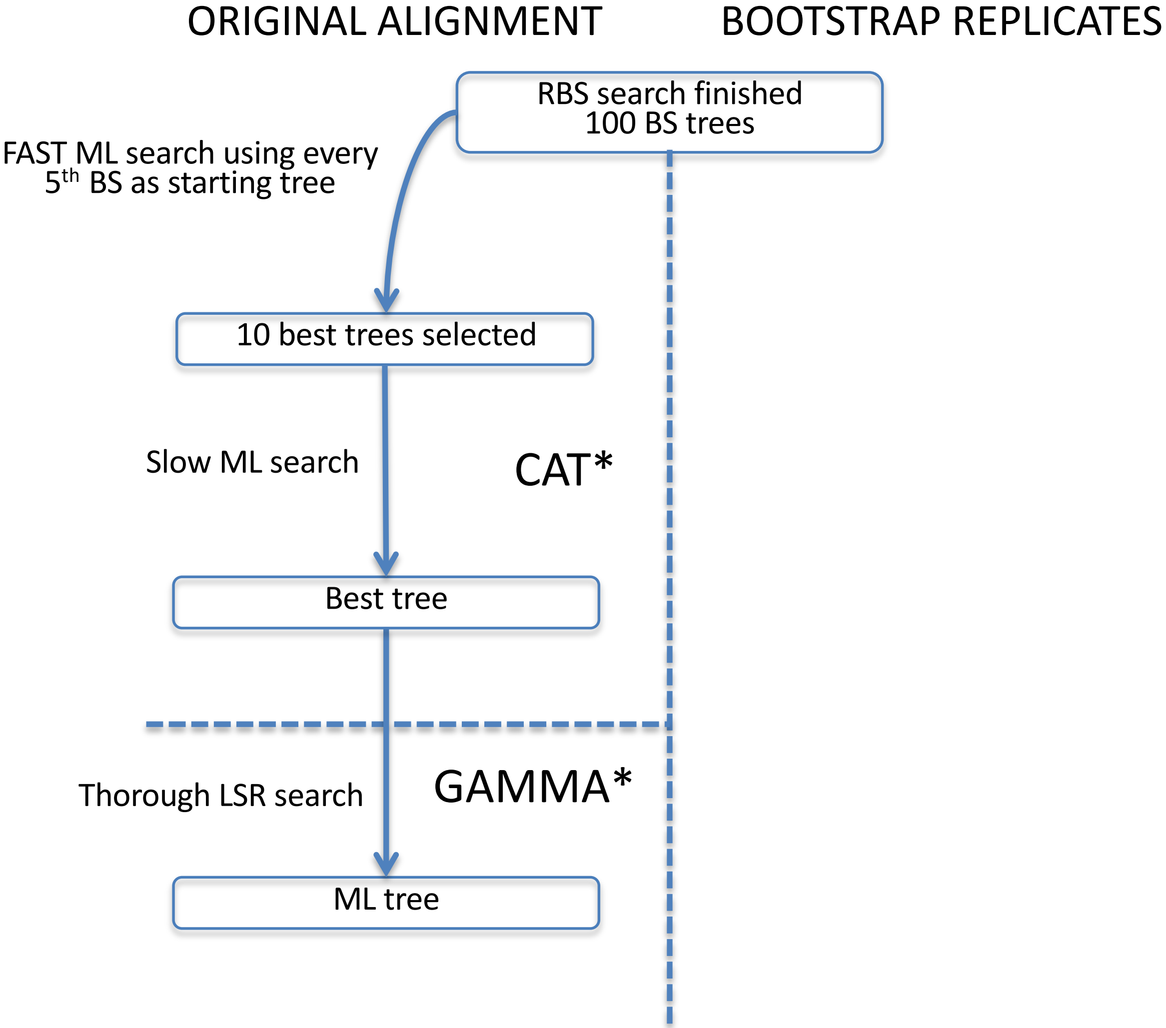
RBS search finished
100 BS trees

1. Model parameters estimated only once
2. Bootstrap trees start from 10 different MP trees (avoid local optima)
3. Dependency in BS tree computation due to using final tree of BS replicate as starting tree for next replicate
4. rough approximation of substitution rate heterogeneity across sites (CAT model)
5. *n* in LSR[1] is set randomly to values between 5 and 15
6. only 2 LSR iterations
7. use a stricter $lh_{cuttoff}$ (50%)
8. evaluate only 5 best trees after LSR
9. speed improvement of ~20x

---

[1] **L**azy **S**ubtree pruning and **R**egrafting

The last step of rapid bootstrapping is the inference of the ML tree



ORIGINAL ALIGNMENT

BOOTSTRAP REPLICATES

RBS search finished
100 BS trees

FAST ML search using every 5th BS as starting tree

10 best trees selected

Slow ML search

CAT*

Best tree

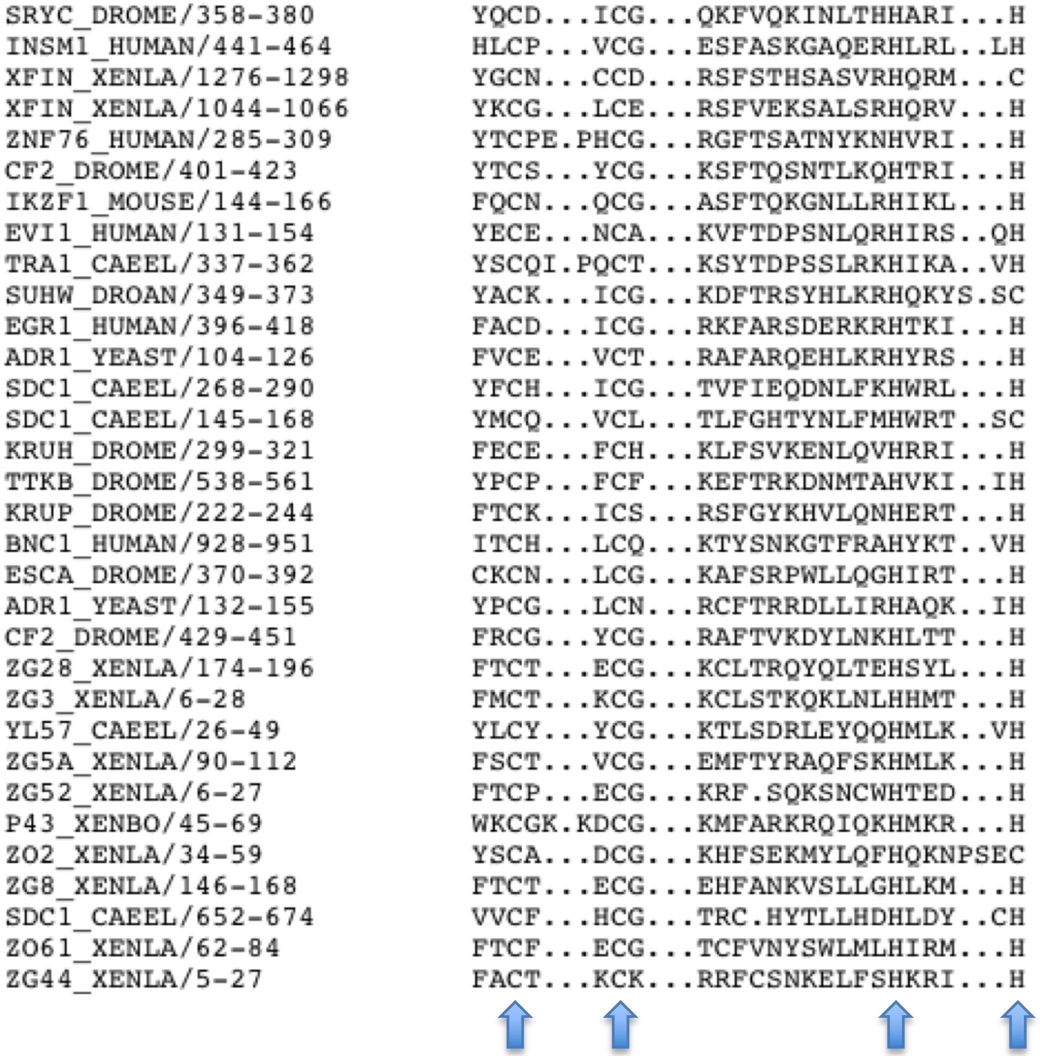
Thorough LSR search

GAMMA*

ML tree

*CAT and GAMMA are two different ways to account for substitution rate heterogeneity across sites where GAMMA is the more complex one

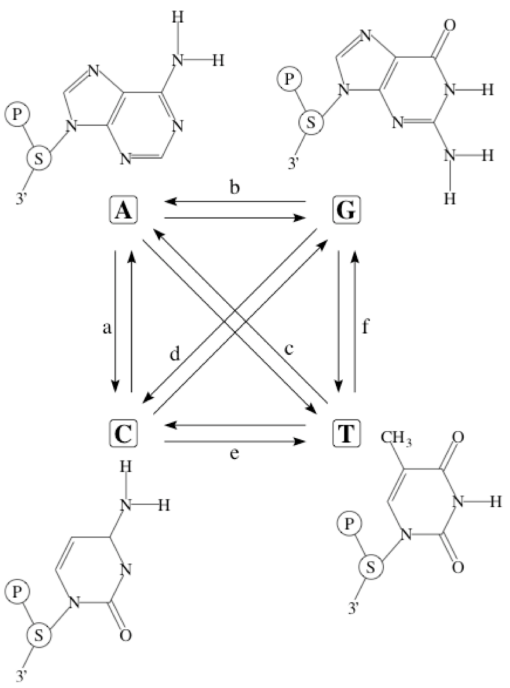
Modeling Substitution rate heterogeneity across sites



```
SRYC_DROME/358-380        YQCD...ICG...QKFVQKINLTHHARI...H
INSM1_HUMAN/441-464       HLCP...VCG...ESFASKGAQERHLRL..LH
XFIN_XENLA/1276-1298      YGCN...CCD...RSFSTHSASVRHQRM...C
XFIN_XENLA/1044-1066      YKCG...LCE...RSFVEKSALSRHQRV...H
ZNF76_HUMAN/285-309       YTCPE.PHCG...RGFTSATNYKNHVRI...H
CF2_DROME/401-423         YTCS...YCG...KSFTQSNTLKQHTRI...H
IKZF1_MOUSE/144-166       FQCN...QCG...ASFTQKGNLLRHIKL...H
EVI1_HUMAN/131-154        YECE...NCA...KVFTDPSNLQRHIRS..QH
TRA1_CAEEL/337-362        YSCQI.PQCT...KSYTDPSSLRKHIKA..VH
SUHW_DROAN/349-373        YACK...ICG...KDFTRSYHLKRHQKYS.SC
EGR1_HUMAN/396-418        FACD...ICG...RKFARSDERKRHTKI...H
ADR1_YEAST/104-126        FVCE...VCT...RAFARQEHLKRHYRS...H
SDC1_CAEEL/268-290        YFCH...ICG...TVFIEQDNLFKHWRL...H
SDC1_CAEEL/145-168        YMCQ...VCL...TLFGHTYNLFMHWRT..SC
KRUH_DROME/299-321        FECE...FCH...KLFSVKENLQVHRRI...H
TTKB_DROME/538-561        YPCP...FCF...KEFTRKDNMTAHVKI..IH
KRUP_DROME/222-244        FTCK...ICS...RSFGYKHVLQNHERT...H
BNC1_HUMAN/928-951        ITCH...LCQ...KTYSNKGTFRAHYKT..VH
ESCA_DROME/370-392        CKCN...LCG...KAFSRPWLLQGHIRT...H
ADR1_YEAST/132-155        YPCG...LCN...RCFTRRDLLIRHAQK..IH
CF2_DROME/429-451         FRCG...YCG...RAFTVKDYLNKHLTT...H
ZG28_XENLA/174-196        FTCT...ECG...KCLTRQYQLTEHSYL...H
ZG3_XENLA/6-28            FMCT...KCG...KCLSTKQKLNLHHMT...H
YL57_CAEEL/26-49          YLCY...YCG...KTLSDRLEYQQHMLK..VH
ZG5A_XENLA/90-112         FSCT...VCG...EMFTYRAQFSKHMLK...H
ZG52_XENLA/6-27           FTCP...ECG...KRF.SQKSNCWHTED...H
P43_XENBO/45-69           WKCGK.KDCG...KMFARKRQIQKHMKR...H
ZO2_XENLA/34-59           YSCA...DCG...KHFSEKMYLQFHQKNPSEC
ZG8_XENLA/146-168         FTCT...ECG...EHFANKVSLLGHLKM...H
SDC1_CAEEL/652-674        VVCF...HCG...TRC.HYTLLHDHLDY..CH
ZO61_XENLA/62-84          FTCF...ECG...TCFVNYSWLMLHIRM...H
ZG44_XENLA/5-27           FACT...KCK...RRFCSNKELFSHKRI...H
```

How to account for conserved/slowly evolving positions in the substitution model?

Modeling rate across sites
Revisiting substitution models



$$Q=\begin{array}{cccc} \mathbf{A} & \mathbf{C} & \mathbf{G} & \mathbf{T} \\ \end{array}$$

$$Q=\begin{pmatrix} - & a & b & c \\ a & - & d & e \\ b & d & - & f \\ c & e & f & - \end{pmatrix}$$

It is a convention to set the diagonal entries q_{ii} such that the rows sum up to 0. Thus,

$$q_{ii}=-\sum_{j\neq i} q_{ij}$$

However, this model assumes that all sites in a sequence, or all columns in an alignment evolve with the same relative rate. Note, that we can rewrite the total rate for a given position as

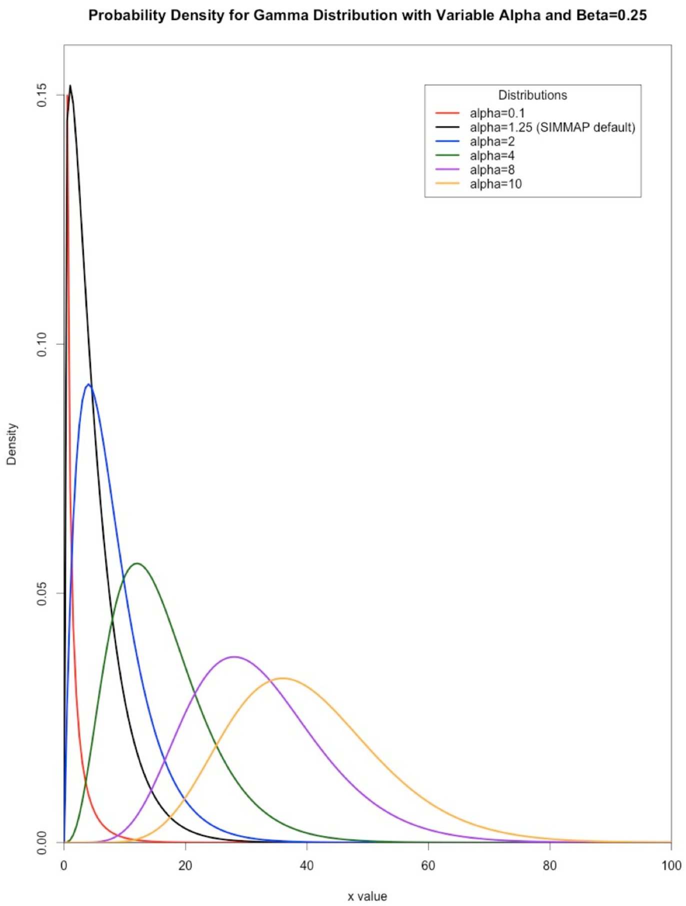
$$q_i=\sum_{j\neq i} q_{ij}$$

We can now introduce a neutral parameter *r=1* such that can re-write q_i as $q_i{}^*r$

For a sequence of L characters we have now the possibility to give the parameter *r* for *l=1…L* a site specific value r_l.

We re-scale the substitution rate in a site-specific manner, i.e. the substitution rate at a position *l* is $q_i r_l$

Modeling rate across sites
Common approaches



Continuous Gamma distribution with a mean of 1*.
Note that the parameter α determines the shape of the distribution.
(Problem of over-parameterization and over-fitting)

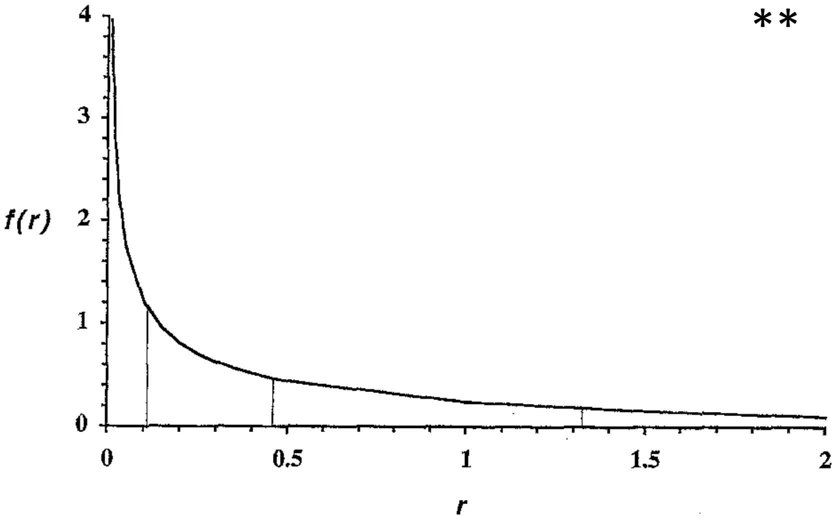
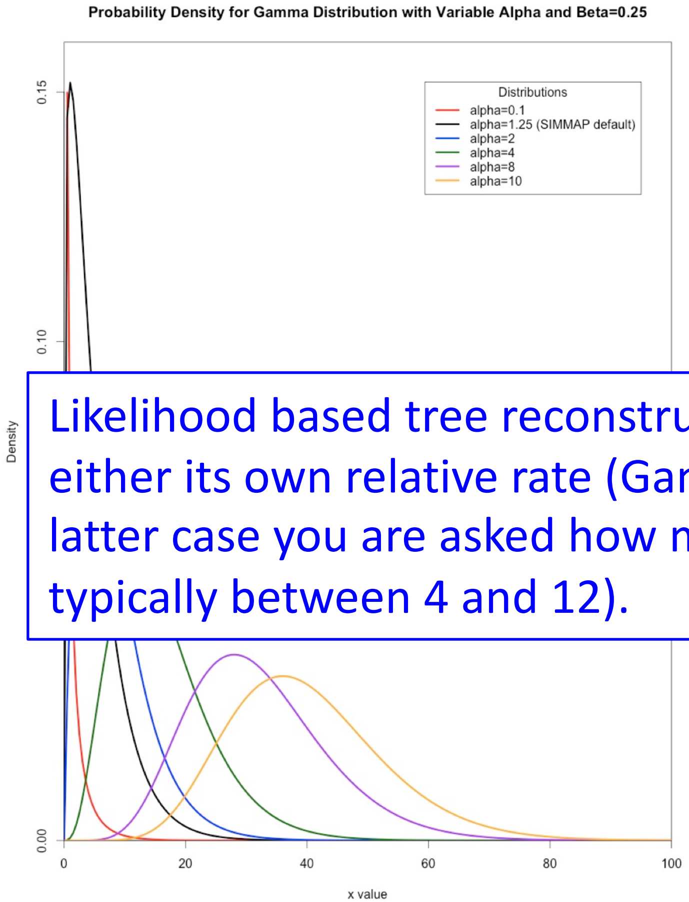


Fig. 1. Discrete approximation to the gamma distribution G(α,β), with $\alpha = \beta = {}^1/_2$. Four categories are used to approximate the continuous distribution, with equal probability for each category. The three boundaries are 0.1015, 0.4549, and 1.3233, which are the percentage points corresponding to $p = {}^1/_4, {}^2/_4, {}^3/_4$. The means of the four categories are 0.0334, 0.2519, 0.8203, 2.8944. The medians are 0.0247, 0.2389, 0.7870, 2.3535, and these are scaled to get 0.0291, 0.2807, 0.9248, and 2.7654, so that the mean of the discrete distribution is one.

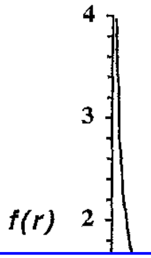
*Thus, if I choose randomly from this distribution, the overall rate will not change!; **Z. Yang (1994) J Mol Evol 39:306

Modeling rate across sites
Common approaches



Continuous Gamma distribution with a mean of 1*.
Note that the parameter α determines the shape of the distribution.
(Problem of over-parameterization and over-fitting)

Likelihood based tree reconstruction methods assign each position in the alignment either its own relative rate (Gamma model) or assigns it to a given rate category. In the latter case you are asked how many rate categories you want to use (values range typically between 4 and 12).

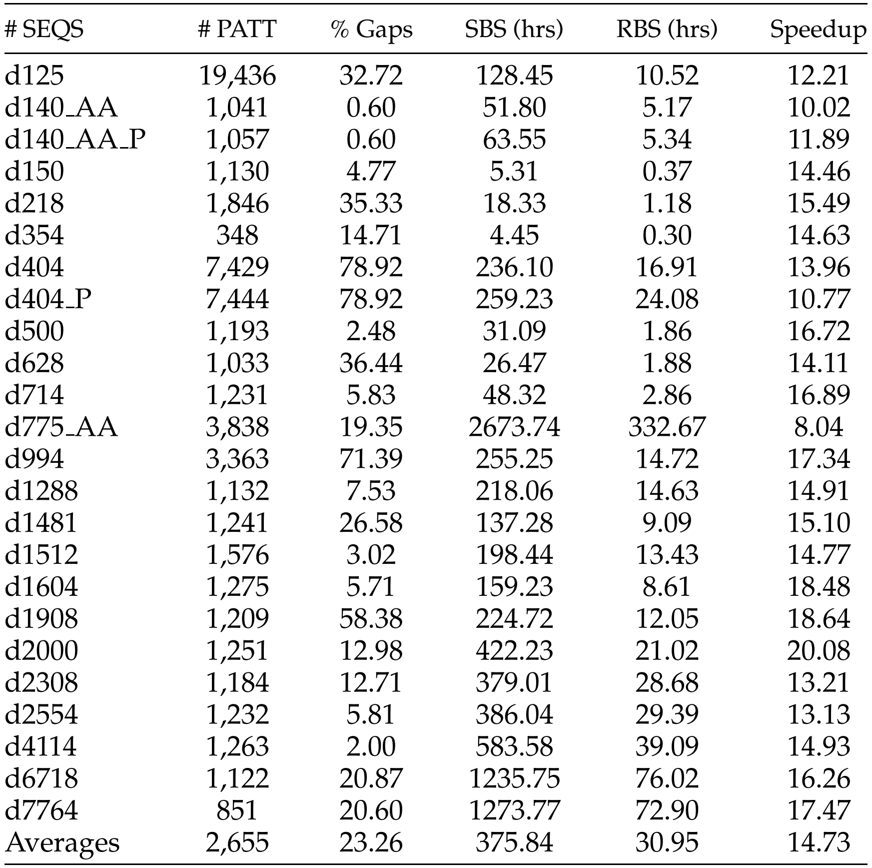


**

Fig. 1. Discrete approximation to the gamma distribution G(α,β), with $\alpha = \beta = {}^1/_2$. Four categories are used to approximate the continuous distribution, with equal probability for each category. The three boundaries are 0.1015, 0.4549, and 1.3233, which are the percentage points corresponding to $p = {}^1/_4, {}^2/_4, {}^3/_4$. The means of the four categories are 0.0334, 0.2519, 0.8203, 2.8944. The medians are 0.0247, 0.2389, 0.7870, 2.3535, and these are scaled to get 0.0291, 0.2807, 0.9248, and 2.7654, so that the mean of the discrete distribution is one.

*Thus, if I choose randomly from this distribution, the overall rate will not change!; **Z. Yang (1994) J Mol Evol 39:306

Approximate speed-up of the Rapid Bootstrap Method



# SEQS	# PATT	% Gaps	SBS (hrs)	RBS (hrs)	Speedup
d125	19,436	32.72	128.45	10.52	12.21
d140_AA	1,041	0.60	51.80	5.17	10.02
d140_AA_P	1,057	0.60	63.55	5.34	11.89
d150	1,130	4.77	5.31	0.37	14.46
d218	1,846	35.33	18.33	1.18	15.49
d354	348	14.71	4.45	0.30	14.63
d404	7,429	78.92	236.10	16.91	13.96
d404_P	7,444	78.92	259.23	24.08	10.77
d500	1,193	2.48	31.09	1.86	16.72
d628	1,033	36.44	26.47	1.88	14.11
d714	1,231	5.83	48.32	2.86	16.89
d775_AA	3,838	19.35	2673.74	332.67	8.04
d994	3,363	71.39	255.25	14.72	17.34
d1288	1,132	7.53	218.06	14.63	14.91
d1481	1,241	26.58	137.28	9.09	15.10
d1512	1,576	3.02	198.44	13.43	14.77
d1604	1,275	5.71	159.23	8.61	18.48
d1908	1,209	58.38	224.72	12.05	18.64
d2000	1,251	12.98	422.23	21.02	20.08
d2308	1,184	12.71	379.01	28.68	13.21
d2554	1,232	5.81	386.04	29.39	13.13
d4114	1,263	2.00	583.58	39.09	14.93
d6718	1,122	20.87	1235.75	76.02	16.26
d7764	851	20.60	1273.77	72.90	17.47
Averages	2,655	23.26	375.84	30.95	14.73

#Seqs: Number of sequences; #PATT: Number distinct patterns; SBS: Standard Bootstrap; RBS: Rapid Bootstrap

Table from Stamatakis et al. 2008 Syst Biol 57:758-771

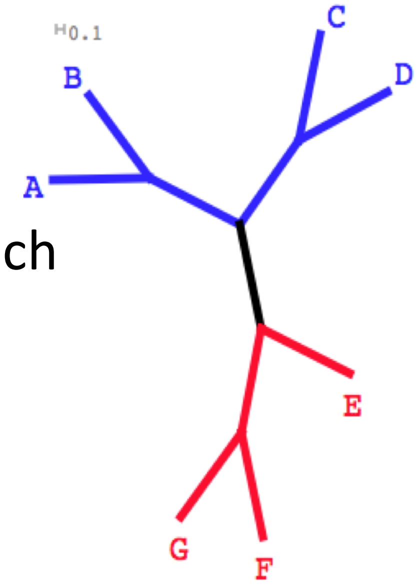
Looking at trees via their **splits**

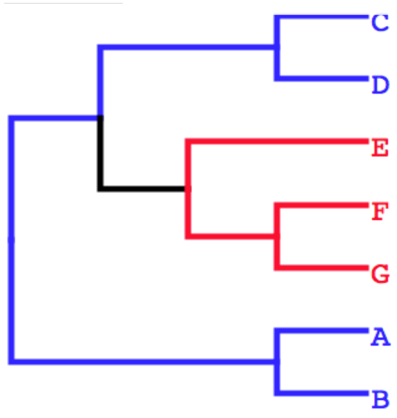
Each branch of a tree describes a **split** of OTUs into two sets

These sets correspond to the two clades associated with the branch

e.g. black branch of the tree specifies the split ABCD | EFG

- can also be written ADCB | GFE etc.
- i.e. the taxon lists in the two halves of the split are unordered





Looking at trees via their **splits**

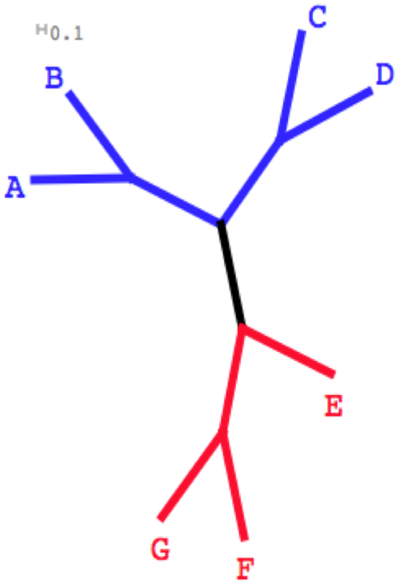
Splits are either

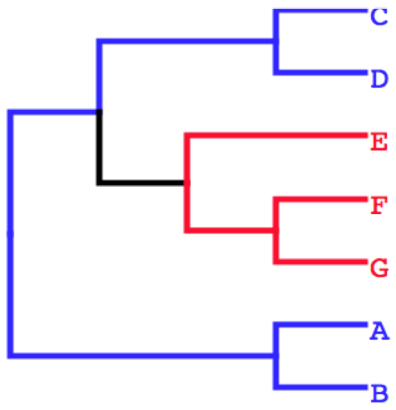
trivial

- example: F | ABCDEG
- associated with **terminal** branches
- provide **no** information about topology structure

non-trivial

- example: ABCD | EFG
- associated with **internal** branches
- provide information about the tree topology

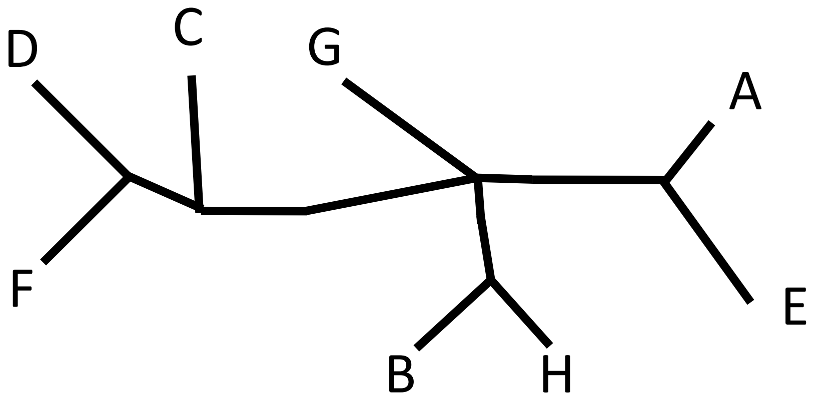




Looking at trees via their **splits**

Complete list of splits described by a tree allows reconstruction of that tree's topology

Helps to consider the sets of clades described by the splits



DF | ABCEGH

BCDFGH | AE

ABEGH | CDF

BH | ACDEFG

Split Compatibility

Sets (e.g. pairs) of splits are either:

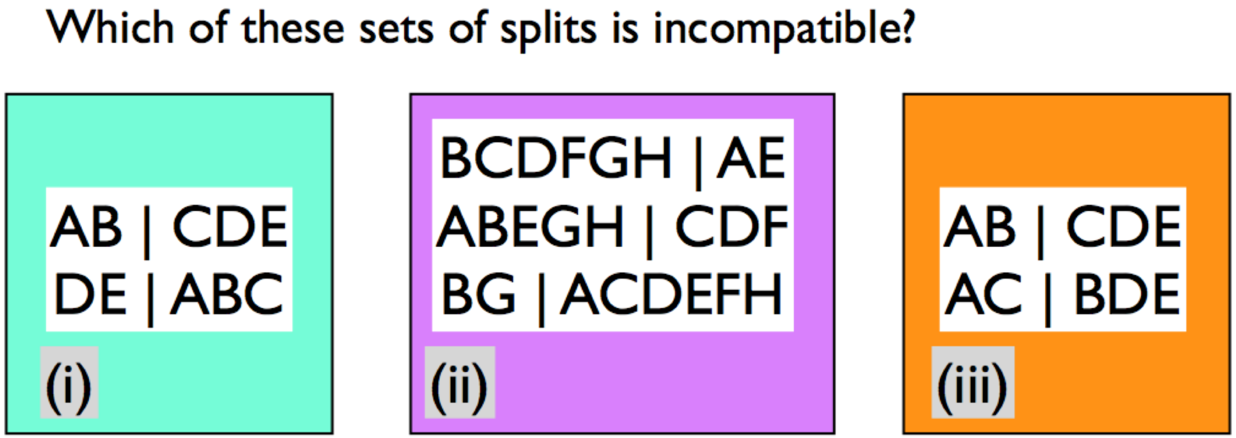
compatible

- a tree **can** be drawn that contains all splits in the set

incompatible

- a tree **cannot** be drawn that contains all splits in the set

Definition: Two splits W|X and Y|Z are compatible, i.e. not contradictory, if at least one intersection of W ∩ Y ,
W ∩ Z , X ∩ Y , X ∩ Z is empty.



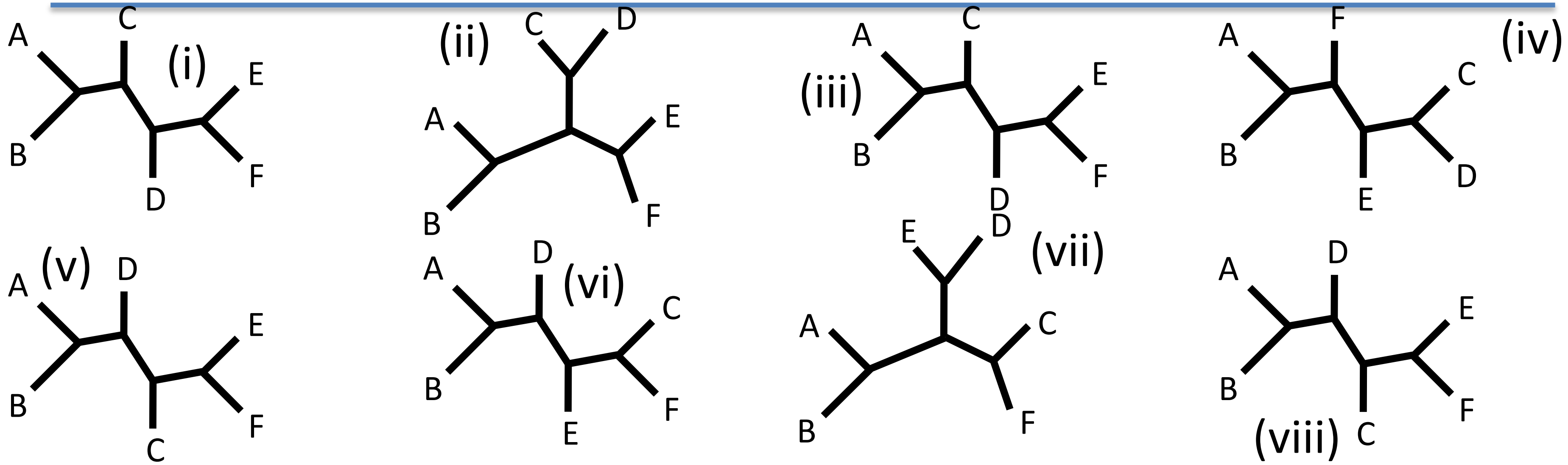
Which of these sets of splits is incompatible?

AB | CDE
DE | ABC
(i)

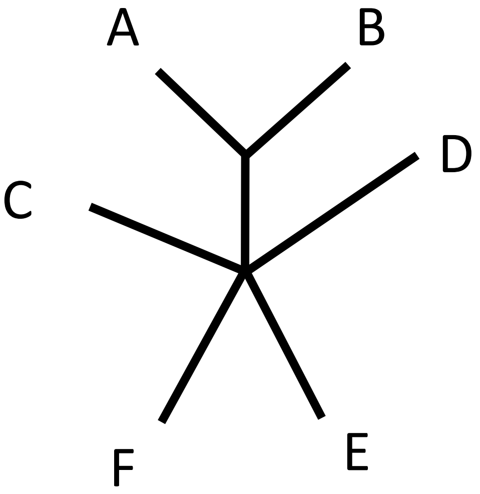
BCDFGH | AE
ABEGH | CDF
BG | ACDEFH
(ii)

AB | CDE
AC | BDE
(iii)

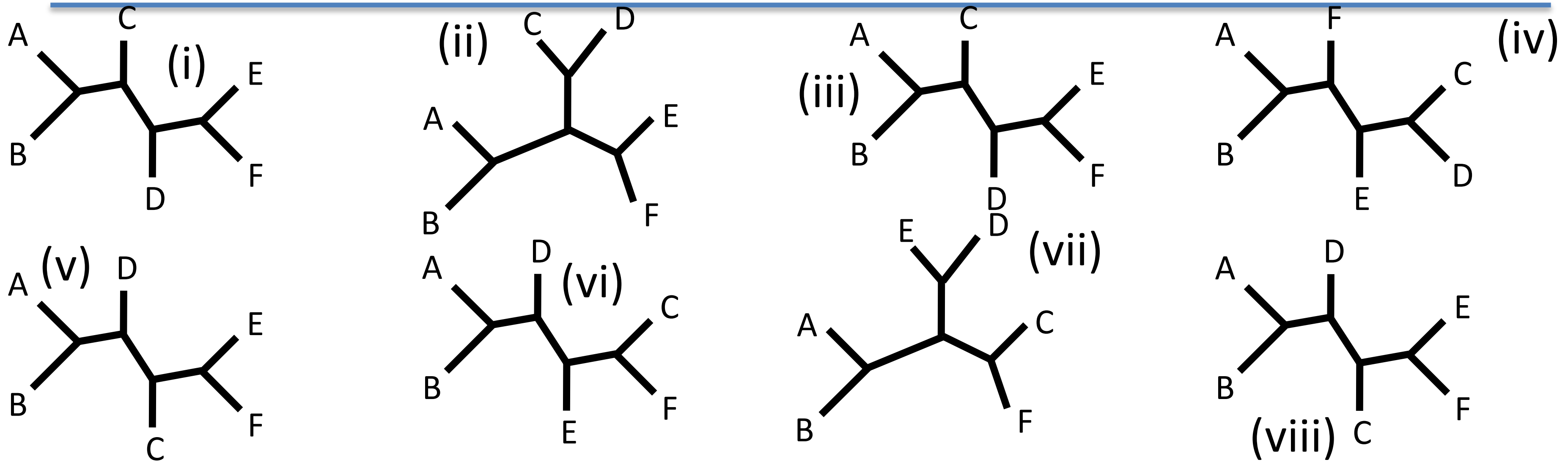
Sets of trees can be summarized by looking at their split sets: Strict Consensus Trees



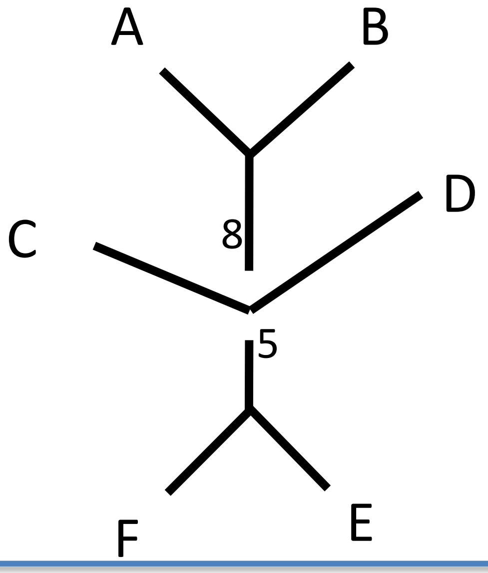
	i	ii	iii	iv	v	vi	vii	viii	
AB | CDEF	*	*	*	*	*	*	*	*	8
CD | ABEF		*		*					2
EF | ABCD	*	*	*		*			*	5
ABC | DEF	*		*						2
DE | ABCF							*		1
CF | ABED						*	*		2
ABD | ECF					*	*		*	3
ABF | CDE				*					1



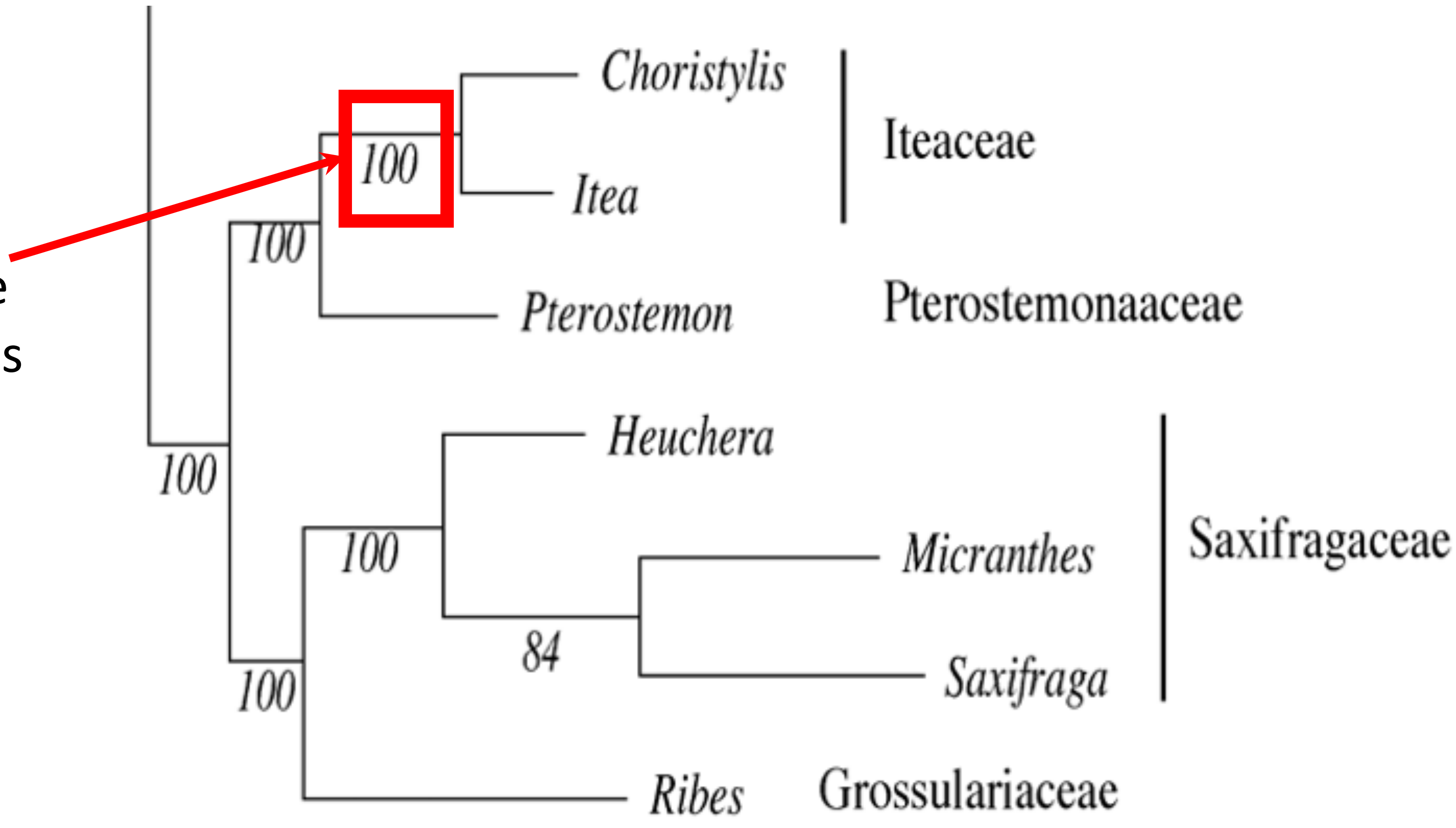
Sets of trees can be summarized by looking at their split sets: 50% Majority Rule Consensus Trees



	i	ii	iii	iv	v	vi	vii	viii	
AB | CDEF	*	*	*	*	*	*	*	*	8
CD | ABEF		*		*					2
EF | ABCD	*	*	*		*			*	5
ABC | DEF	*		*						2
DE | ABCF							*		1
CF | ABED						*	*		2
ABD | ECF					*	*		*	3
ABF | CDE				*					1



Label the Branches!



Branches of consensus tree labeled to indicate proportion of trees containing that branch/split

Resolving an ancient, rapid radiation in Saxifragales.
Jian S, Soltis PS, Gitzendanner MA, Moore MJ, Li R, Hendry TA, Qiu YL, Dhingra A, Bell CD, Soltis DE.
Syst Biol. 2008 Feb;57(1):38-57.
PMID: 18275001

The missing bit: Tree evaluation using Bayes theorem

So far we have computed

$$P(D \mid \mathrm{T},\Theta)$$

i.e. the likelihood of the data *D* given the tree *T* and the parameter vector Θ.

However, what we are interested in most of the times is the likelihood of T and Θ given D, i.e.

$$P(\mathrm{T},\Theta \mid D)$$

The missing bit: Tree evaluation using Bayes theorem

So far we have computed

$$P(D \mid \mathrm{T}, \Theta)$$

i.e. the likelihood of the data *D* given the tree *T* and the parameter vector *Θ*.

However, what we are interested in most of the times is the likelihood of *T* and *Θ* given *D*, which is given by Bayes' theorem

$$P(\mathrm{T}, \Theta \mid D) = \frac{P(D \mid \mathrm{T}, \Theta) * P(\mathrm{T}, \Theta)}{P(D)}$$

total probability of the data considering all hypotheses. This is the problematic bit!

prior information on the probability of a given hypothesis (T, Θ)

Posterior probabilities from Bayesian tree searches and ML bootstrap values have different meanings!