ML phylogenetic inference and GARLI

Derrick Zwickl

University of Arizona

(and University of Kansas)

Workshop on Molecular Evolution 2015

Outline

- Heuristics and tree searches
- ML phylogeny inference and GARLI
- Using GARLI
- Gap (indel) models

Finding the tree with the best score

Difficulties:

1. **Enormous number of trees to consider**
2. Multiple local optima
3. Nested search: for each tree we should maximize the likelihood:
   - Numerical parameters of the model of sequence evolution
   - Branch-length parameters
   - Optimal parameter values are strongly correlated
   - http://phylo.bio.ku.edu/mephytis/brlen-opt.html
   - http://phylo.bio.ku.edu/mephytis/tree-opt.html

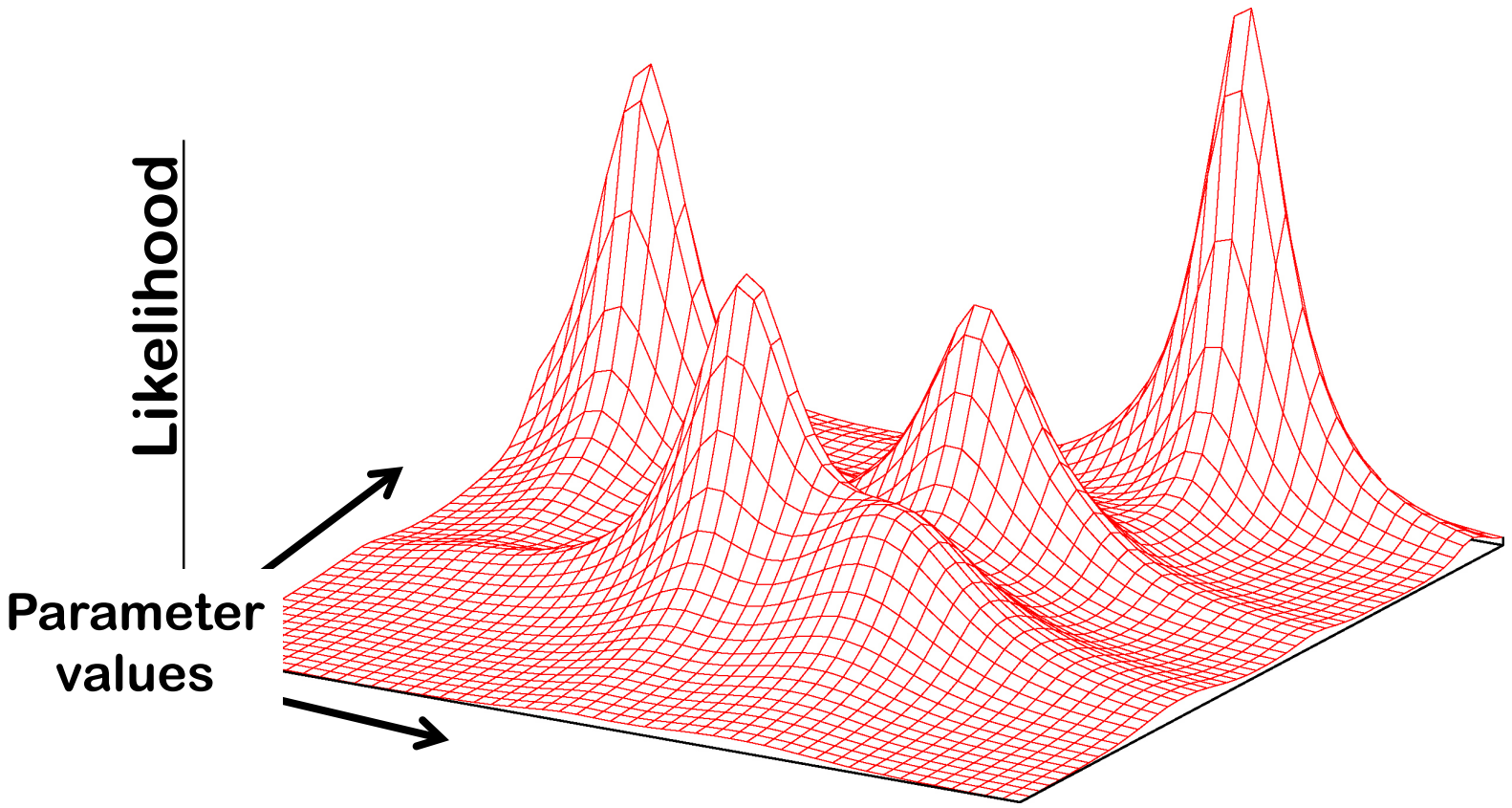
Tips	Number of unrooted (binary) trees	
4	3	
5	15	
6	105	
7	945	
8	10,395	
9	135,135	
10	2,027,025	
11	34,459,425	
12	654,729,075	
13	13,749,310,575	
14	316,234,143,225	
15	7,905,853,580,625	
16	213,458,046,676,875	
17	6,190,283,353,629,375	
18	191,898,783,962,510,625	
19	6,332,659,870,762,850,625	
20	22,164,309,5476,699,771,875	
21	8,200,794,532,637,891,559,375	
22	319,830,986,772,877,770,815,625	
23	13,113,070,457,687,988,603,440,625	> 21 moles of trees
24	563,862,029,680,583,509,947,946,875	

Finding the tree with the best score

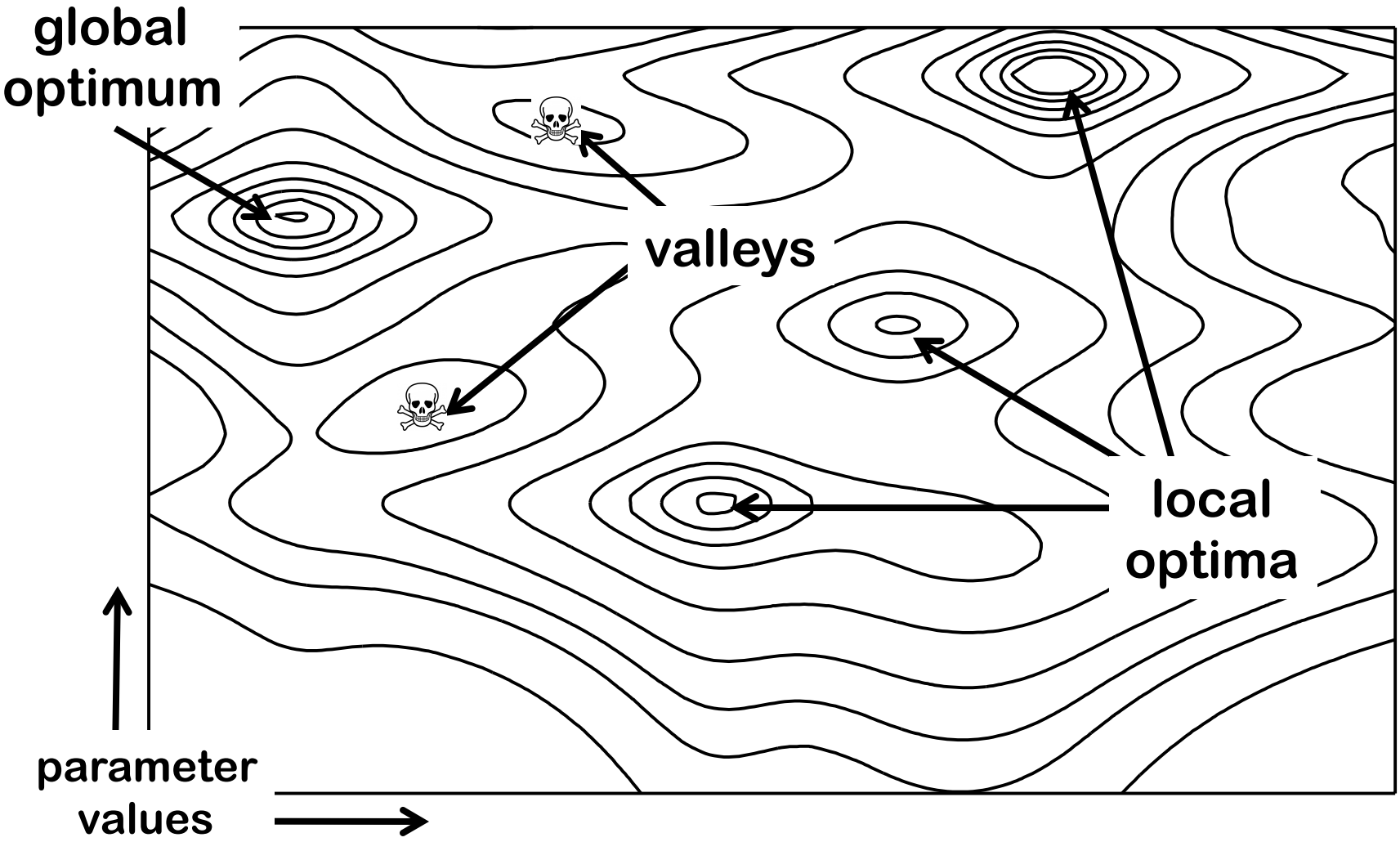
Difficulties:

1. Enormous number of trees to consider
2. **Multiple local optima**
3. Nested search: for each tree we should maximize the likelihood:
   - Numerical parameters of the model of sequence evolution
   - Branch-length parameters
   - Optimal parameter values are strongly correlated
   - http://phylo.bio.ku.edu/mephytis/brlen-opt.html
   - http://phylo.bio.ku.edu/mephytis/tree-opt.html

A likelihood surface



A likelihood surface (from above)



Finding the tree with the best score

Difficulties:

1. Enormous number of trees to consider
2. Multiple local optima
3. Nested search: for each tree we should maximize the likelihood:
   - Numerical parameters of the model of sequence evolution
   - Branch-length parameters
   - Optimal parameter values are strongly correlated
   - http://phylo.bio.ku.edu/mephytis/brlen-opt.html
   - http://phylo.bio.ku.edu/mephytis/tree-opt.html

General heuristic tree search

1. Generate a starting tree. Score this current tree.
2. Look at trees that are “close” to the current tree
3. Possibly update the “current tree”
4. Go back to step 2, unless the search has run a long time without improving the tree score.

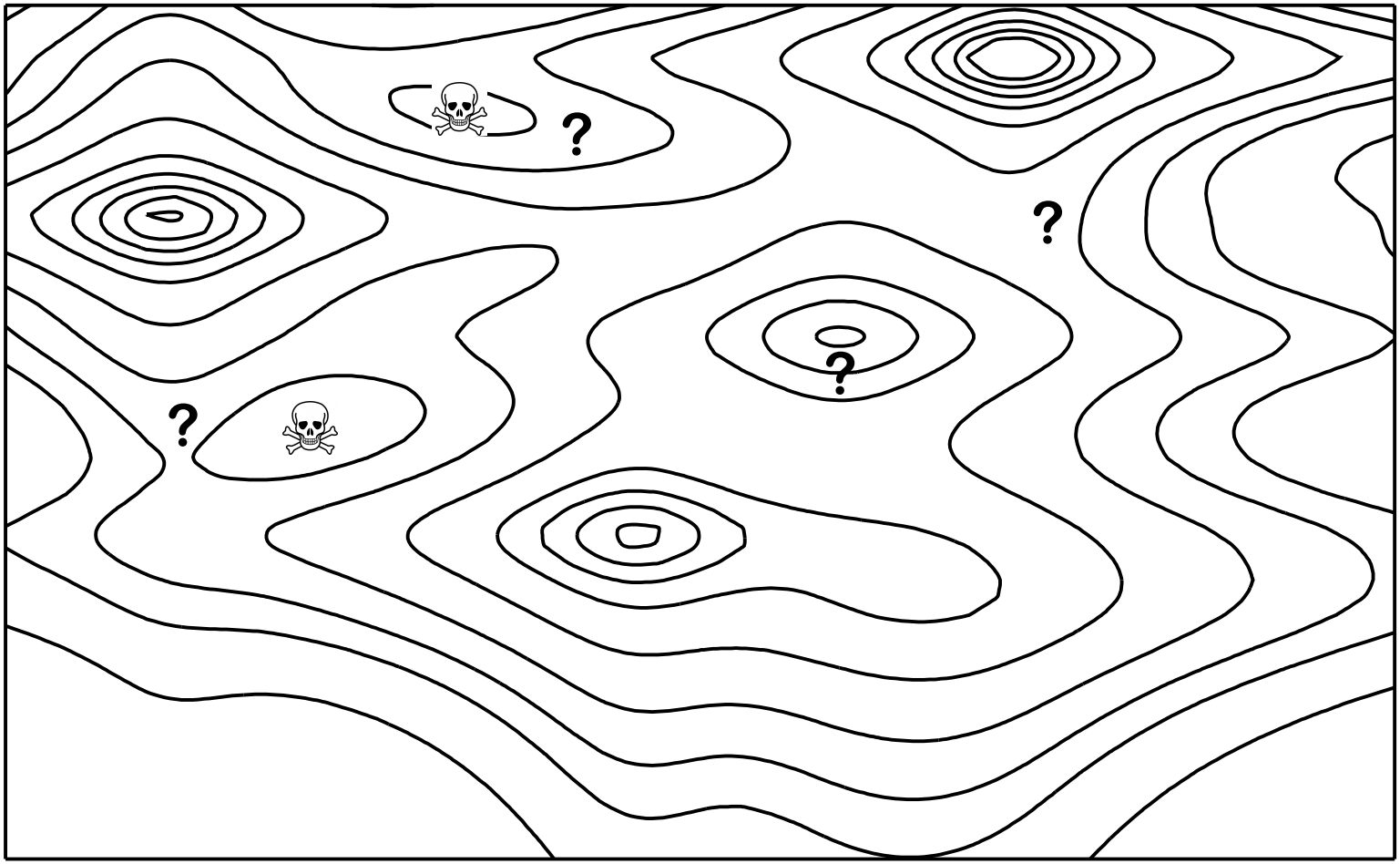
Heuristic search features

1. Where does it start?
2. How are new values proposed ? i.e. what do we mean when we say “look at a tree that is close to the current tree”?
3. How do we decide to accept a proposed tree so that it is the “current tree”?
4. When can you terminate the search?

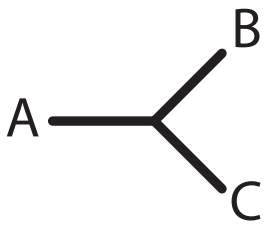
Heuristic features

1. **Where does it start?**
   - **User supplied starting tree**
   - **Star decomposition or Stepwise Addition**
   - **A randomly chosen tree**
   - **…**

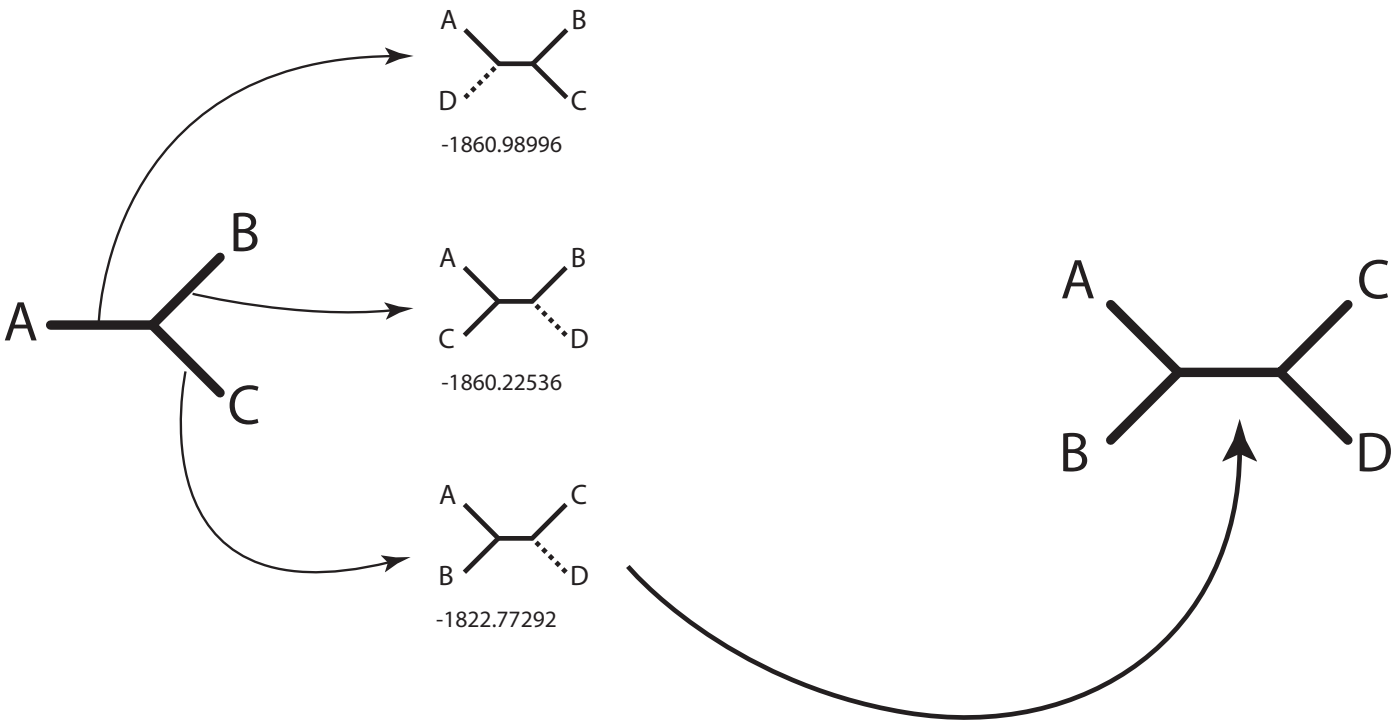
Heuristics: starting point



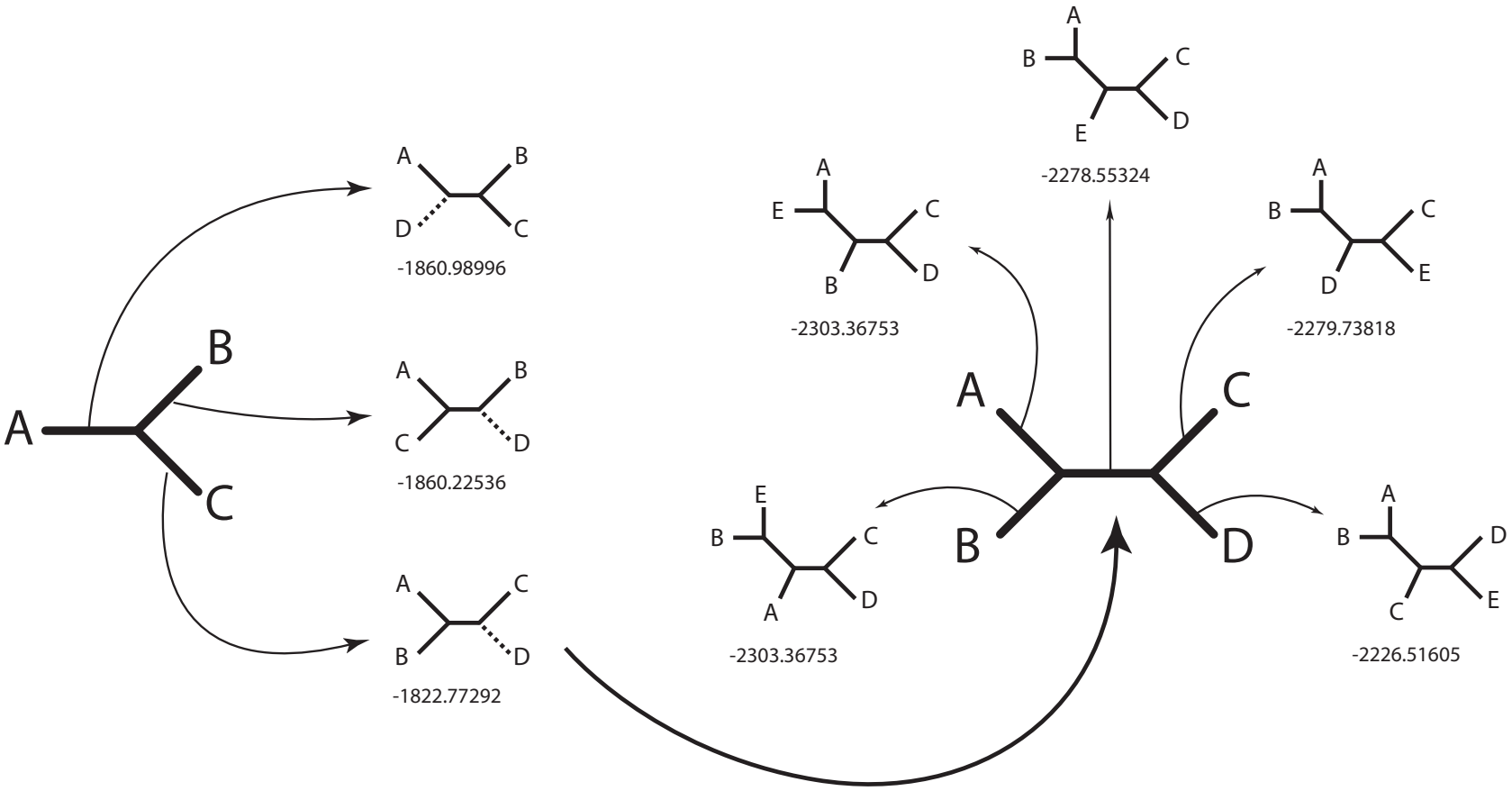
Stepwise addition



Stepwise addition



Stepwise addition



Stepwise addition

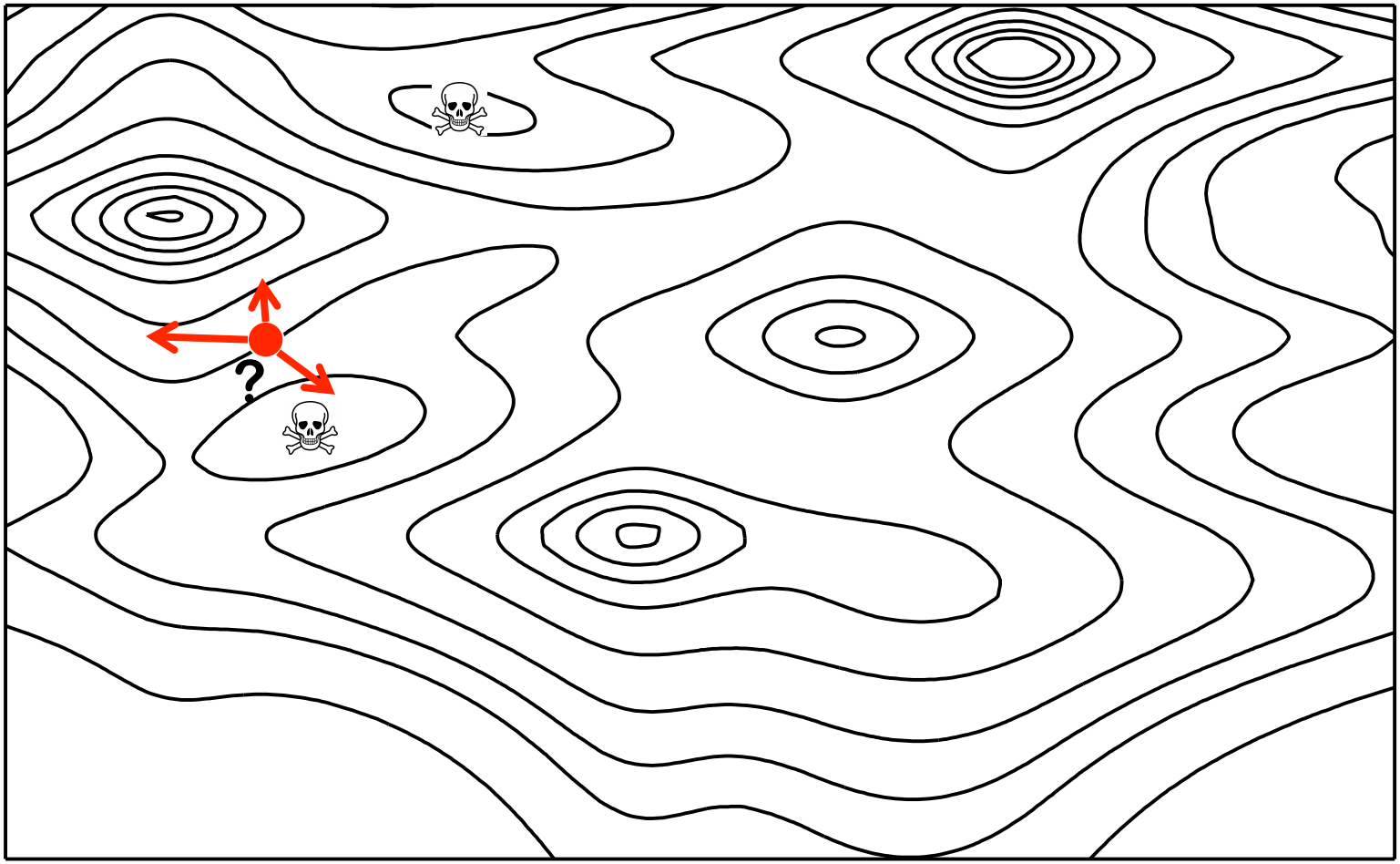
- Greedy, but can introduce a new taxon on the path between taxa that have already been joined.
- The tree can depend on the input order of the taxa
- Number of trees scored for N taxa :

$$\begin{aligned} \#\text{ trees scored} &= \sum_{i=3}^{N-1} (2i-3) \\ &= (N-1)(N-3) \end{aligned}$$

Thus, stepwise addition is $O(N^2)$. For N=10:

$$63 = 3 + 5 + 7 + 9 + 11 + 13 + 15$$

Heuristics: proposing new values



Phylogenetic searches

Think about moving through an abstract "treespace"

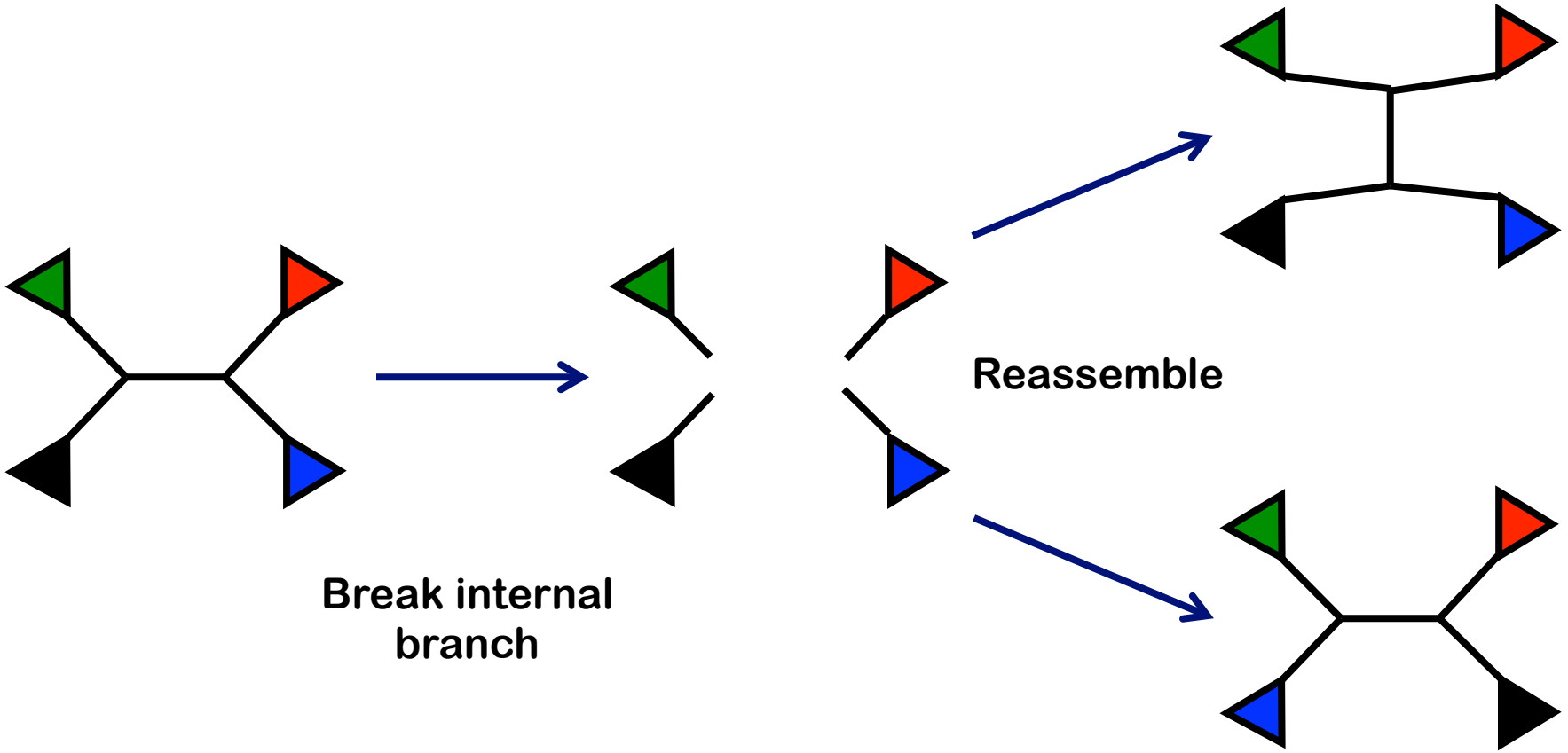
Nearby points in this treespace are connected by NNI (nearest neighbor interchange) branch swaps

Heuristics

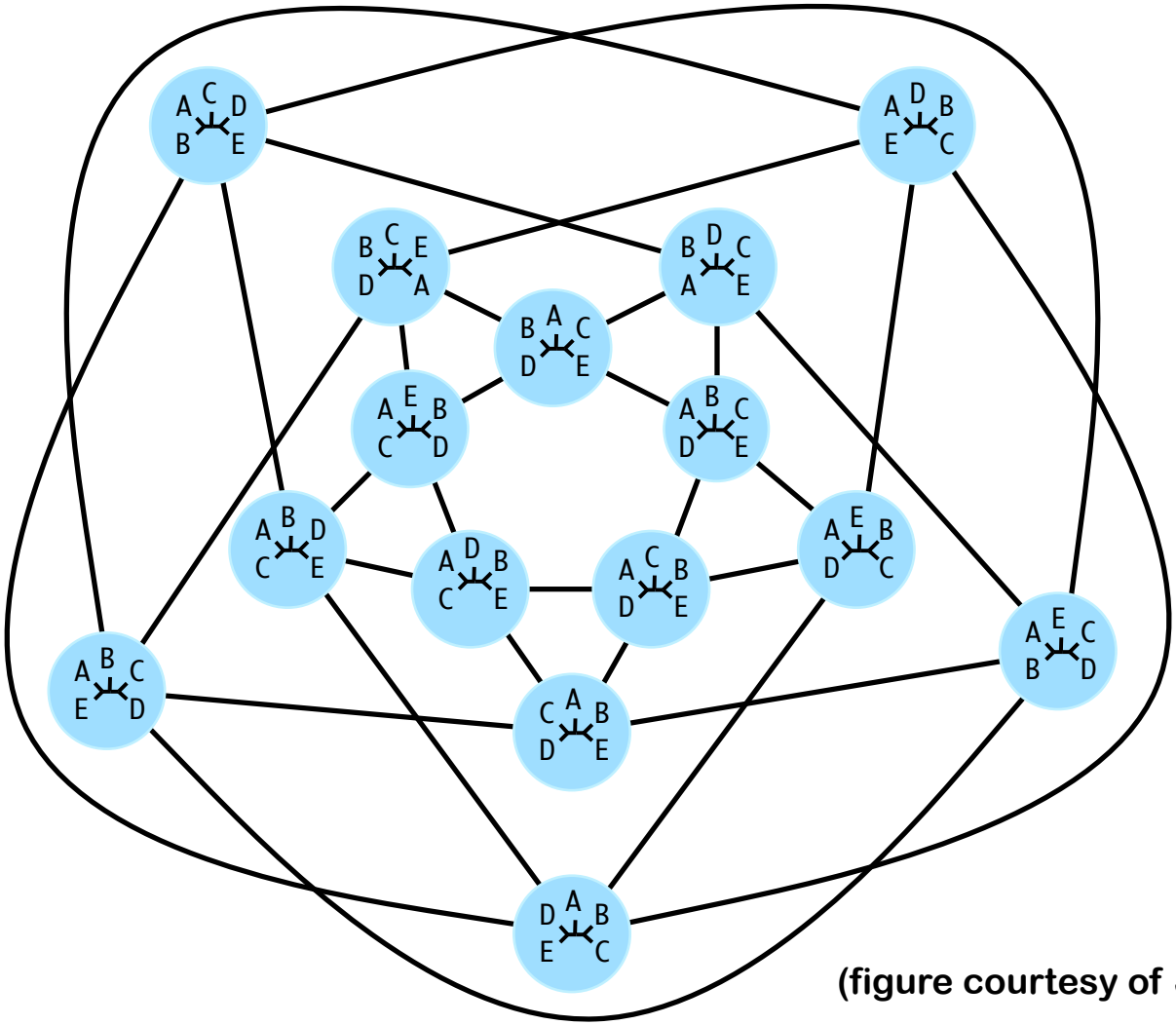
Few restrictions on how a heuristic can work

Best choice likely problem specific

Moving through treespace: NNI branch swaps

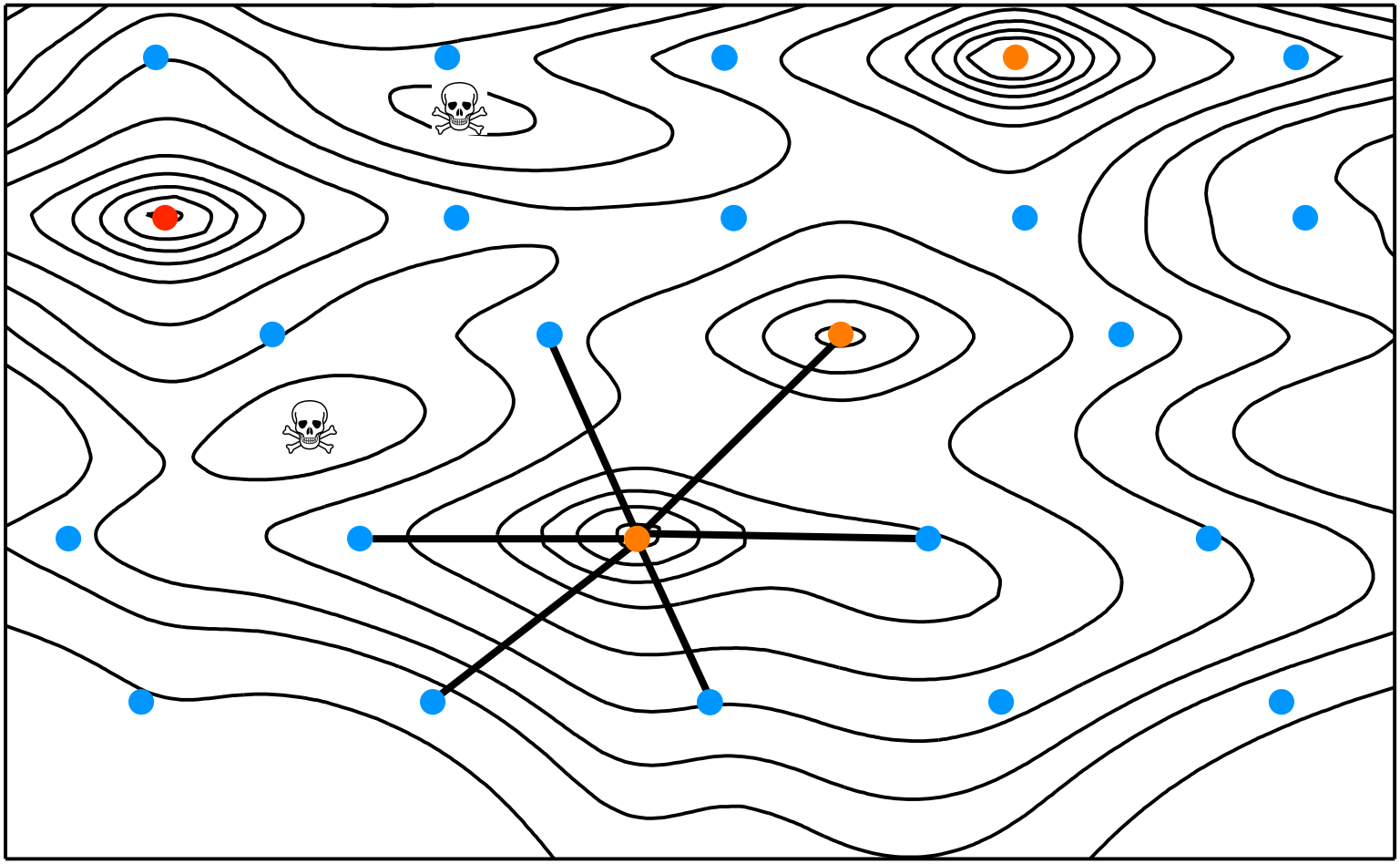


Schoenberg graph – edges connect NNI neighbors

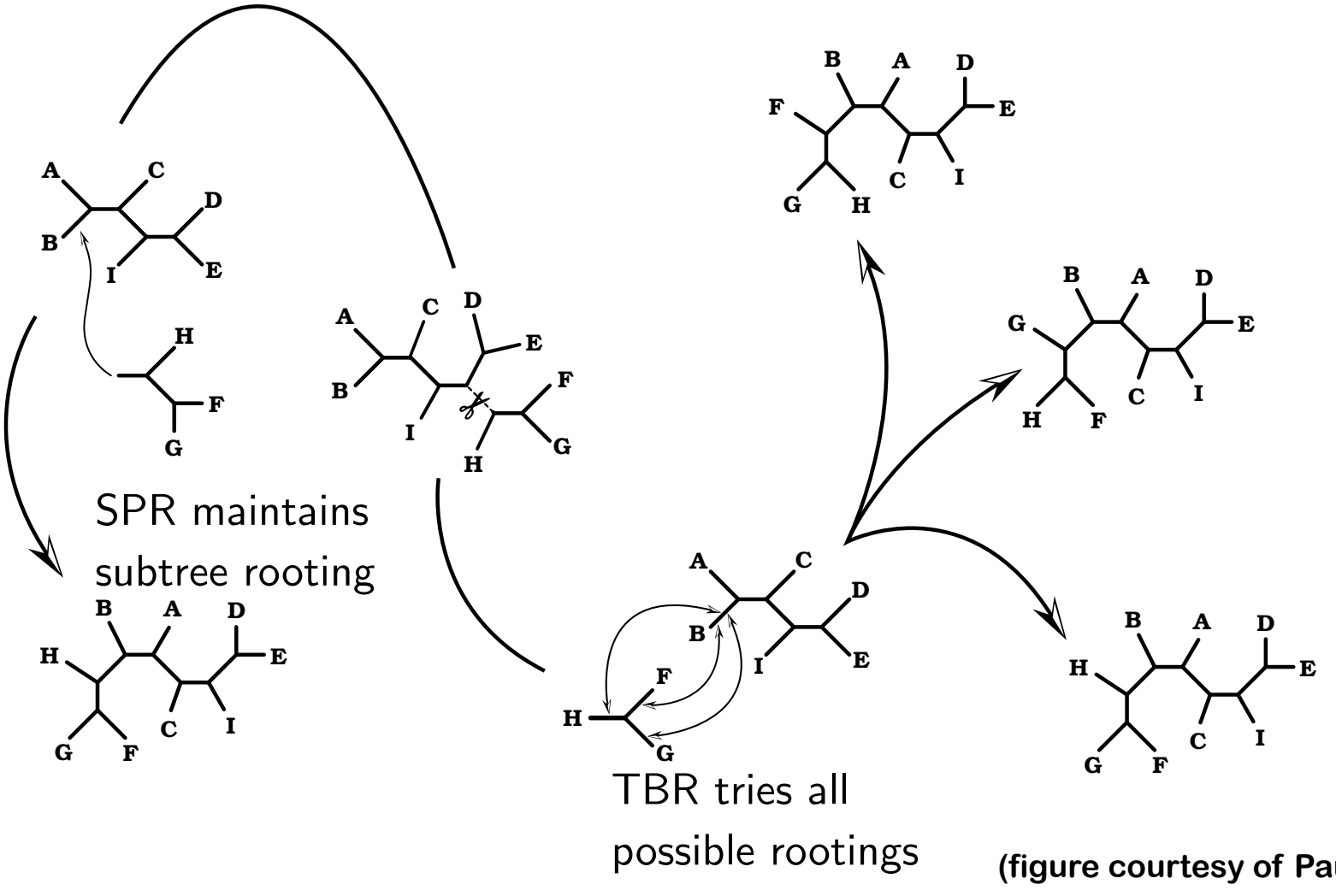


(figure courtesy of Joe Felsenstein)

NNI Treespace

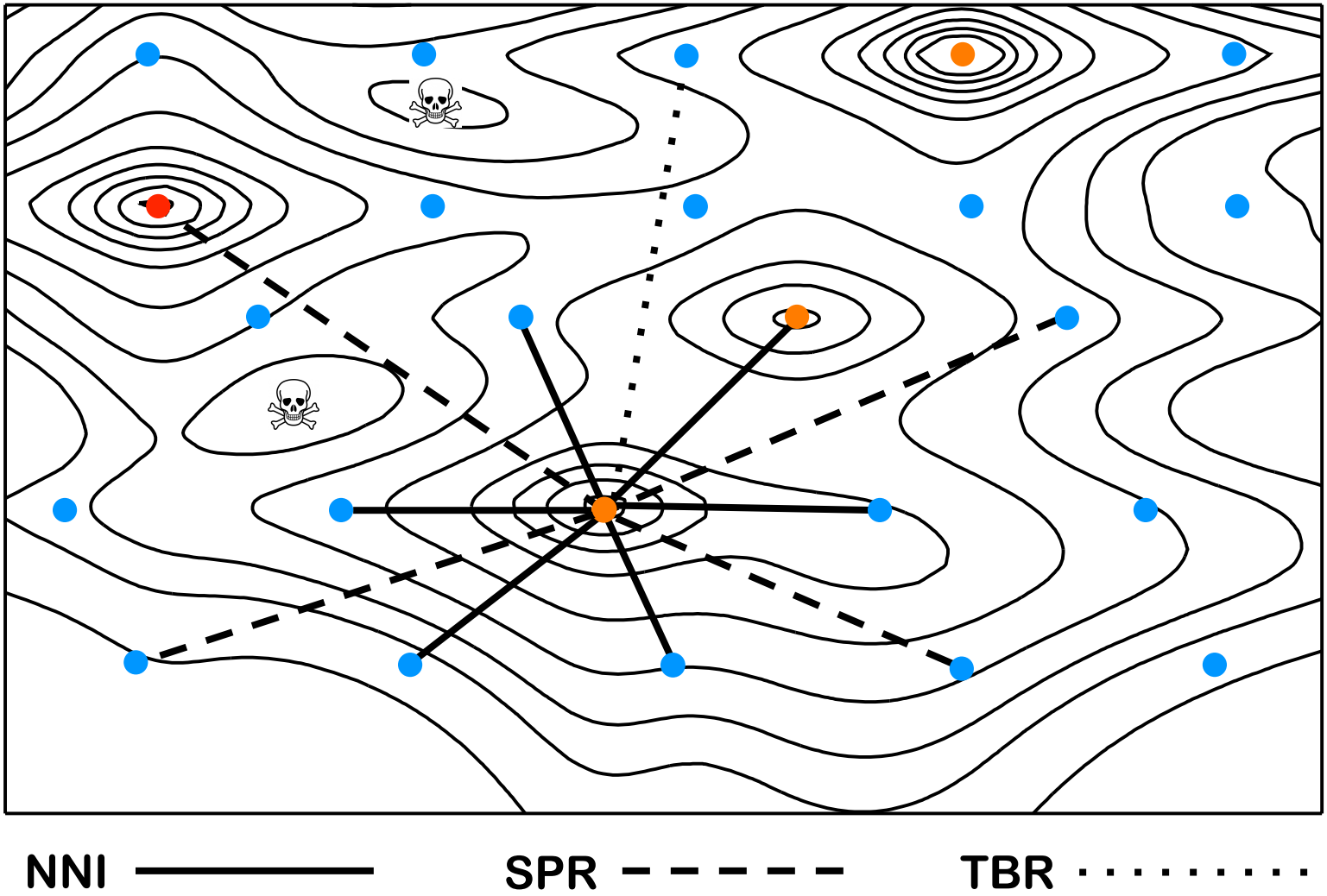


Subtree Pruning Regrafting (SPR) and Tree Bisection Reconnection (TBR)



(figure courtesy of Paul Lewis)

SPR/TBR moves in NNI treespace



PAUP* HSearch – a hill climber

1. User chooses a starting tree method and a branch-swapping operation.
2. Propose a neighboring tree according to the swap
3. Accept the proposed tree if the score is better
4. Terminate if you have looked at every neighbor of the current tree.

GARLI

- Genetic Algorithm for Rapid Likelihood Inference
- Descendent of GAML (Lewis, 1998)
- Stochastic, genetic algorithm-like approach instead of deterministic hill climbing
- Terminates when the score/tree has not changed in a long time.
- Gradually optimizes tree topology, branch lengths and model parameters
- Accurate ML tree inference on large datasets (hundreds of sequences) in hours

The Genetic Algorithm

Computational analog of evolution by natural selection

A few simple requirements:

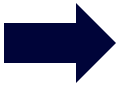
- **Measure of fitness**
- **Method of selection**
- **Mutation operators**
- **Recombination operators**

GA terminology

- Individual
  (topology+model parameter values+branch lengths)
- Population
- Fitness (log-likelihood)
- Selection function (fitness proportional)
- Generation

One generation

Create initial population of individuals



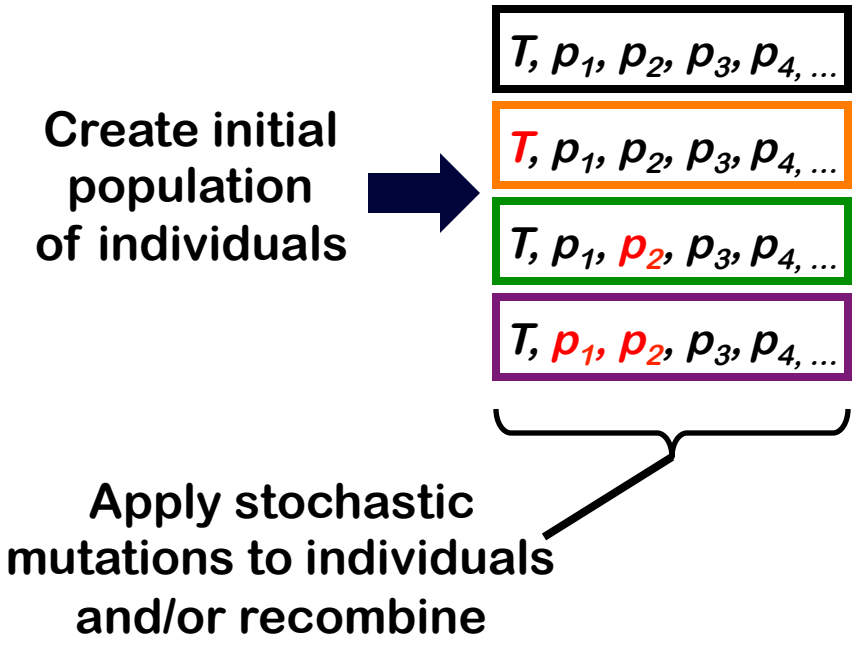
$T, p_1, p_2, p_3, p_4, \ldots$

$T, p_1, p_2, p_3, p_4, \ldots$

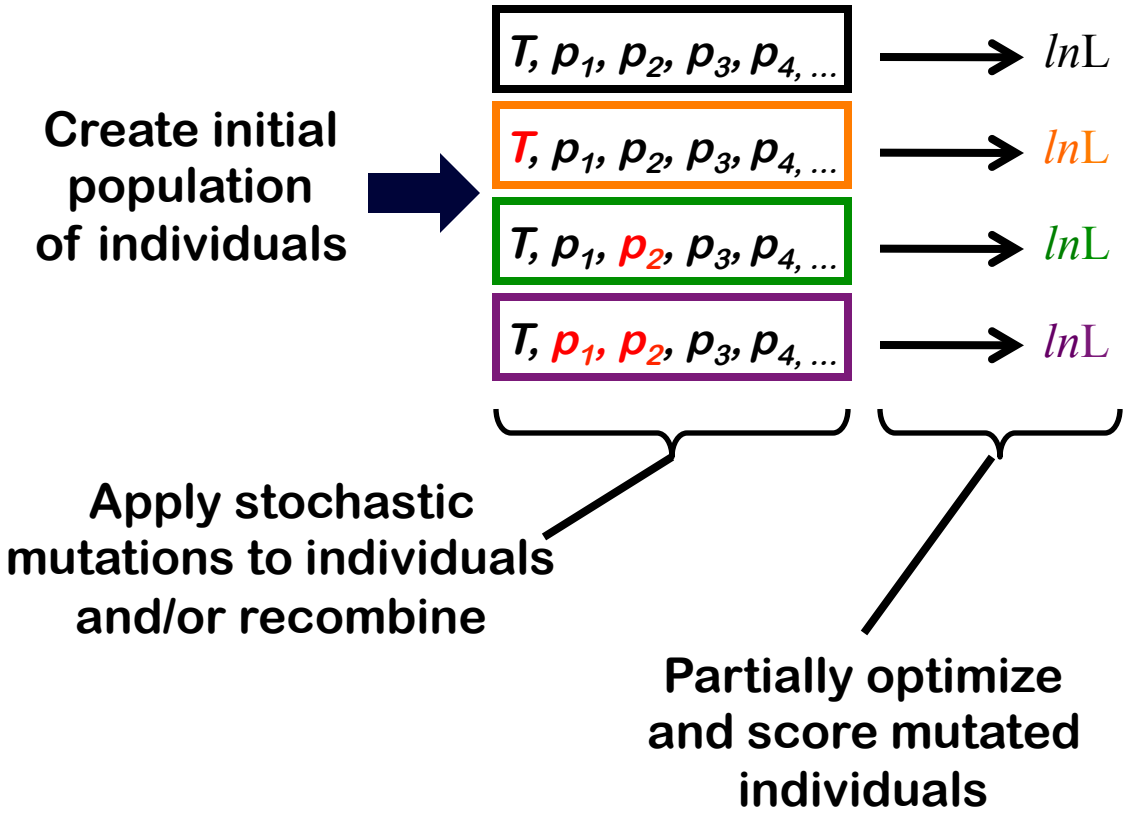
$T, p_1, p_2, p_3, p_4, \ldots$

$T, p_1, p_2, p_3, p_4, \ldots$

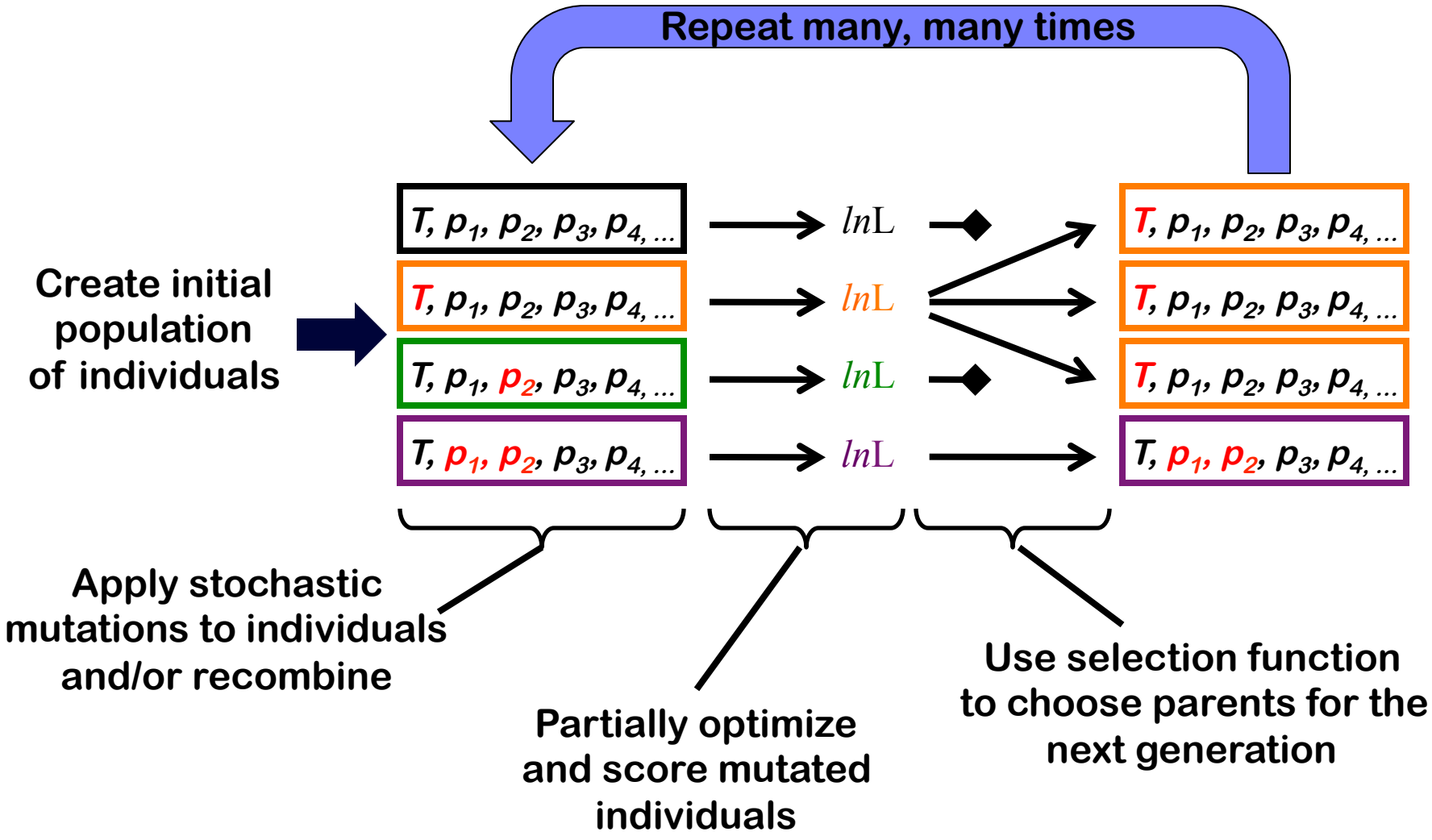
One generation



One generation



One generation



GA mutations

Similar to proposals in Bayesian MCMC phylogenetic methods (but with fewer restrictions)

Random component

GARLI uses independent mutations of tree topology, model parameters and branch-length parameters

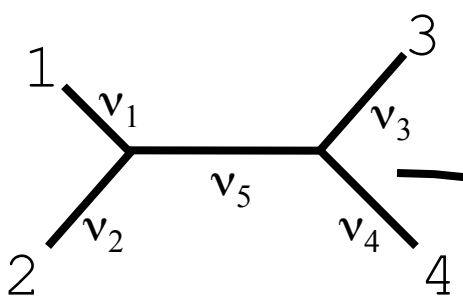
Is GARLI a GA?

Well, no …

The serial algorithm does not use recombination between individuals, which technically disqualifies it

Also atypical of GA's in other aspects (small population sizes, very strong selection pressure)

Computing the likelihood of a topology



Branch-length parameters

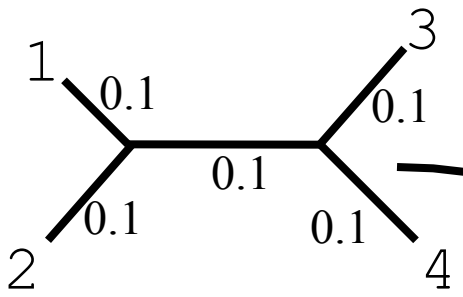
Tree topology

$$\ln L = \ln Pr(X \mid T, \nu_1, \nu_2, \nu_3, \nu_4, \ldots a, b, c \ldots)$$

	A	C	G	T
A	-	$a\pi_C$	$b\pi_G$	$c\pi_T$
C	$a\pi_A$	-	$d\pi_G$	$e\pi_T$
G	$b\pi_A$	$d\pi_C$	-	$f\pi_T$
T	$c\pi_A$	$e\pi_C$	$f\pi_G$	-

Substitution model parameters

Computing the likelihood of a topology



Branch-length parameters

Tree topology

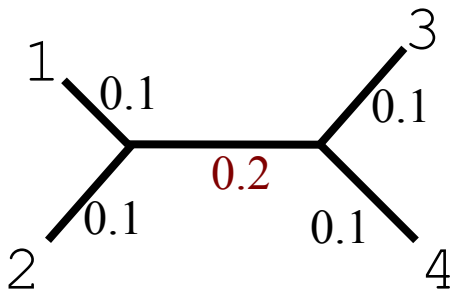
$$\ln L = \ln Pr(X \mid T, 0.1, 0.1, 0.1, 0.1, \ldots 1.0, 2.0, 3.0 \ldots)$$

	A	C	G	T
A	-	1.0	2.0	3.0
C	1.0	-	4.0	5.0
G	2.0	4.0	-	1.0
T	3.0	5.0	1.0	-

Substitution model parameters

= -242.83

Computing the likelihood of a topology



$$\ln L = \ln Pr(X \mid T, 0.1, 0.1, \mathbf{0.2}, 0.1, \ldots 1.0, 2.0, 3.0 \ldots)$$

	A	C	G	T
A	-	1.0	2.0	3.0
C	1.0	-	4.0	5.0
G	2.0	4.0	-	1.0
T	3.0	5.0	1.0	-

= -241.52

The likelihood of a topology

A single topology represents range of likelihood scores, depending on parameter values

The likelihood of a given tree and specific parameter values can be calculated quickly

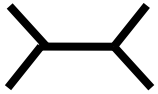
The maximized likelihood of a topology

The "likelihood of a tree": the likelihood with other parameters at their optimal values *for that tree* ("maximized likelihood")

All of the nice statistical theory for ML assumes that we have maximized the likelihood.

This following discussion will mainly deal with optimization of branch-length parameters

Obtaining the maximized likelihood



***ln*L = *f*(T, 0.1, 0.1, 0.2, 0.1 … 1.0, 2.0, 3.0 …)** = -242.83

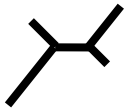
***ln*L = *f*(T, 0.1, 0.1, 0.08, 0.1 … 1.0, 2.0, 3.0 …)** = -241.52

***ln*L = *f*(T, 0.05, 0.1, 0.08, 0.1 … 1.0, 2.0, 3.0 …)** = -241.23

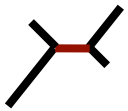
***ln*L = *f*(T, 0.05, 0.1, 0.08, 0.16 … 1.0, 2.0, 3.0 …)** = -241.11

.

.



***ln*L = *f*(T, 0.05, 0.23, 0.08, 0.16 … 1.2, 4.6, 1.7 …)** = -239.31



***ln*L = *f*(T, 0.05, 0.23, 0.12, 0.16 … 1.2, 4.6, 1.7 …)** = -239.29

etc.

Parameter optimization

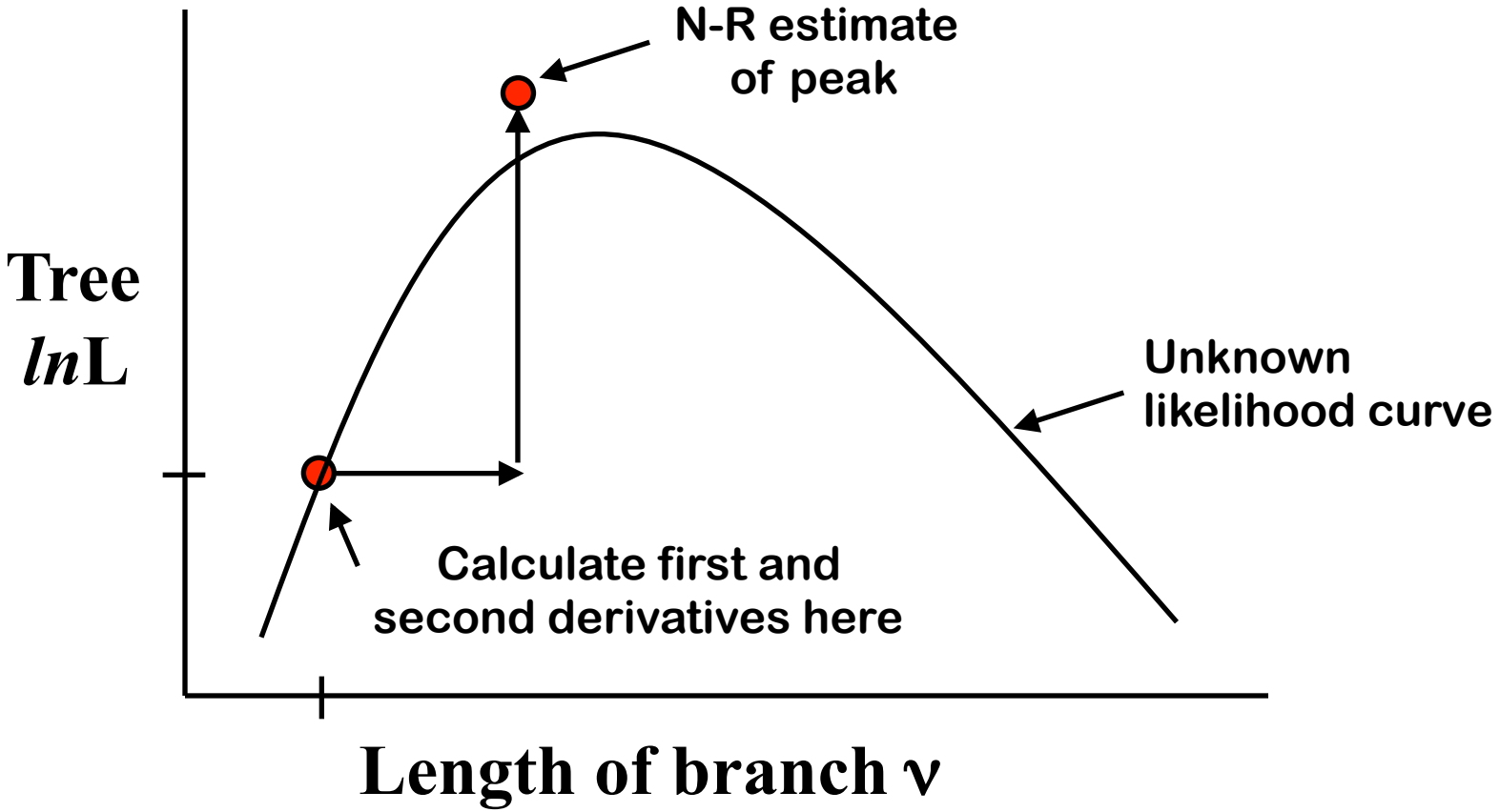
Find the numerical value of a parameter that maximizes the likelihood (conditional on current values of other parameters)

Many methods, generally based on slope of likelihood surface or bracketing of maximum

Parameters often cycled through sequentially

http://phylo.bio.ku.edu/mephytis/brlen-opt.html

Branch-length optimization
(Newton-Raphson method)



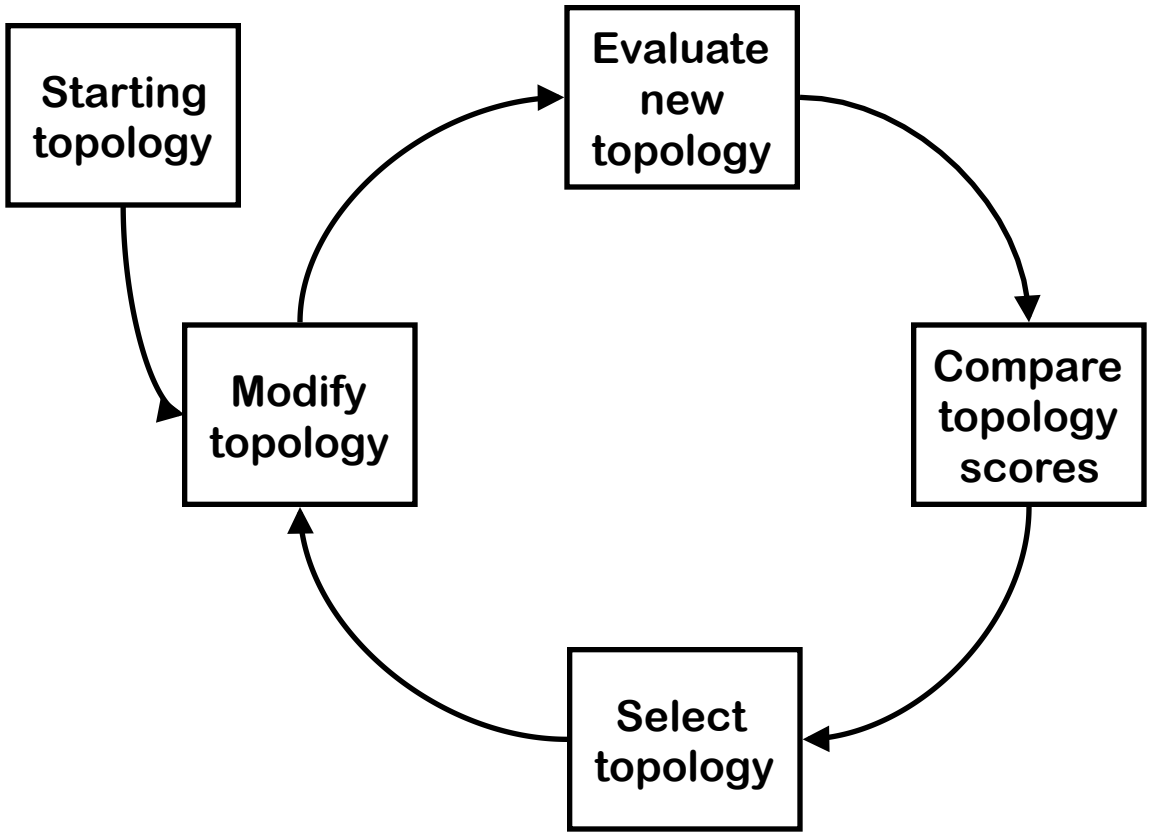
Maximized likelihood: cons

Fully optimizing a single branch length can require significant computation

When one parameter changes, optimal values of all others also change

Heuristic runtimes

Inference time = # of topologies to evaluate X time to evaluate each



Both are strongly a function of the # of sequences when calculating maximized likelihood

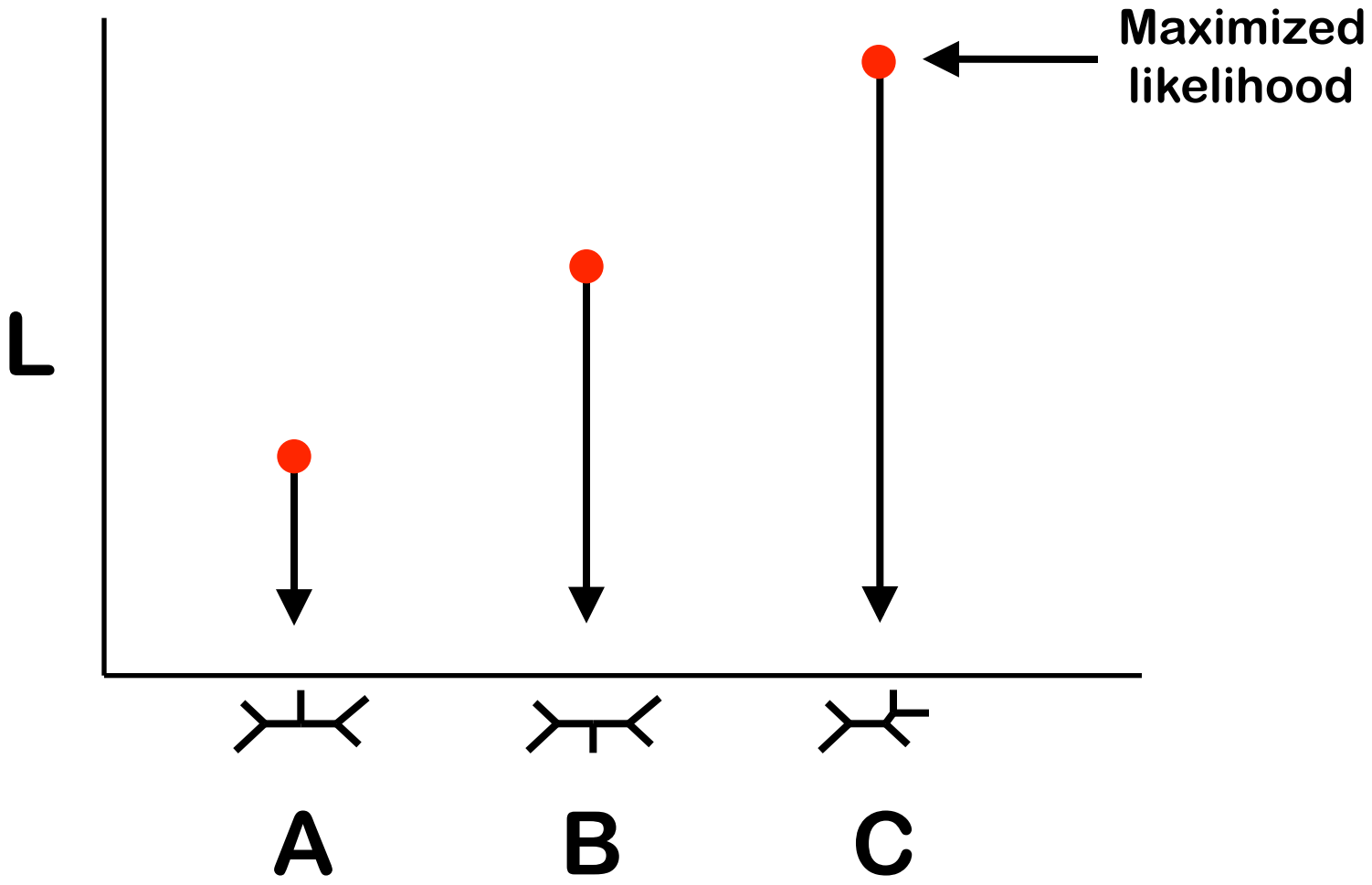
Avoiding the maximized likelihood

We want to accurately judge the merits of topologies, *as if* we had the maximized likelihood

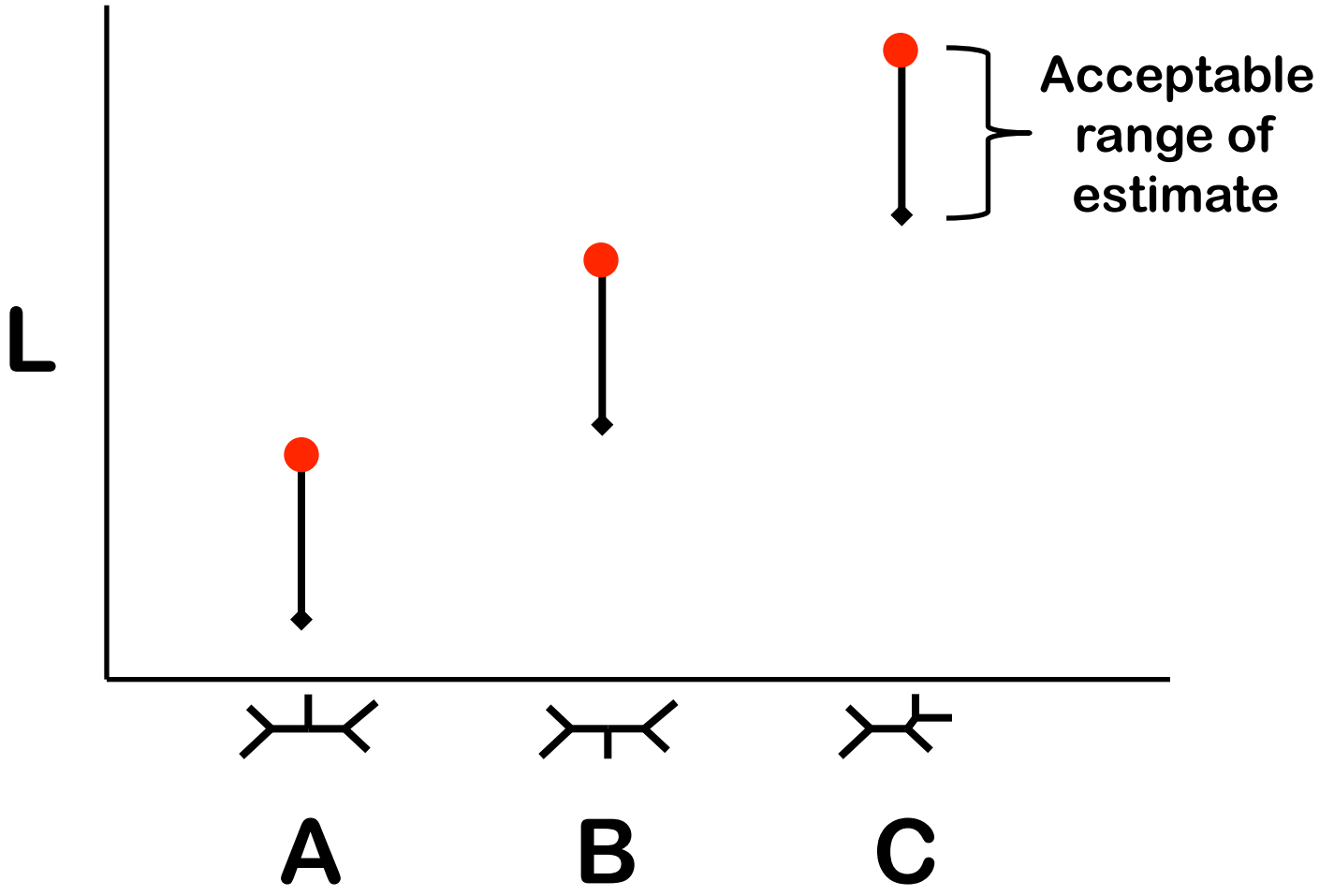
… *but without actually calculating it*

We'll explore the idea of an approximate likelihood score for topologies

How accurate does a tree likelihood estimate need to be?

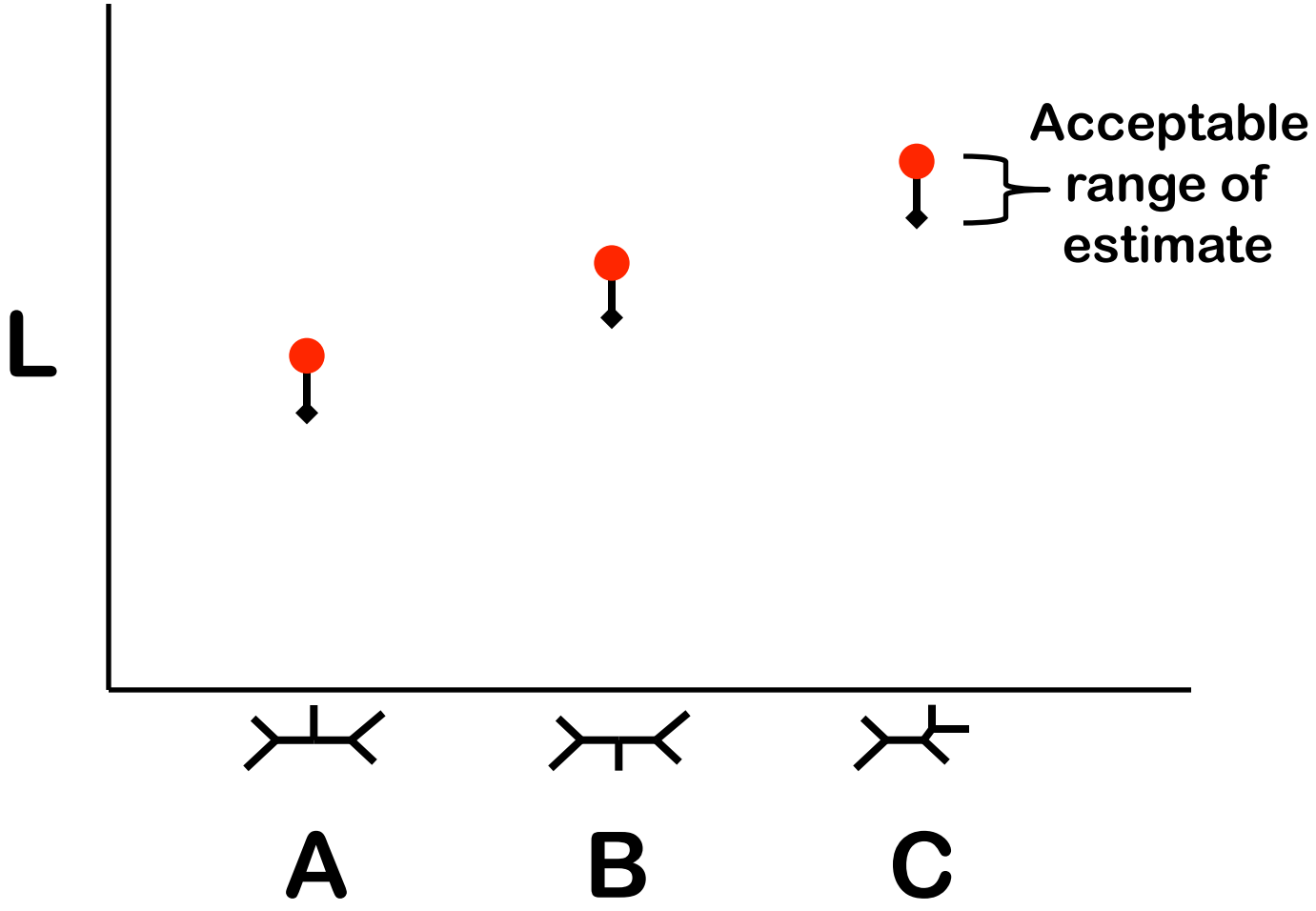


How accurate does a tree likelihood estimate need to be?



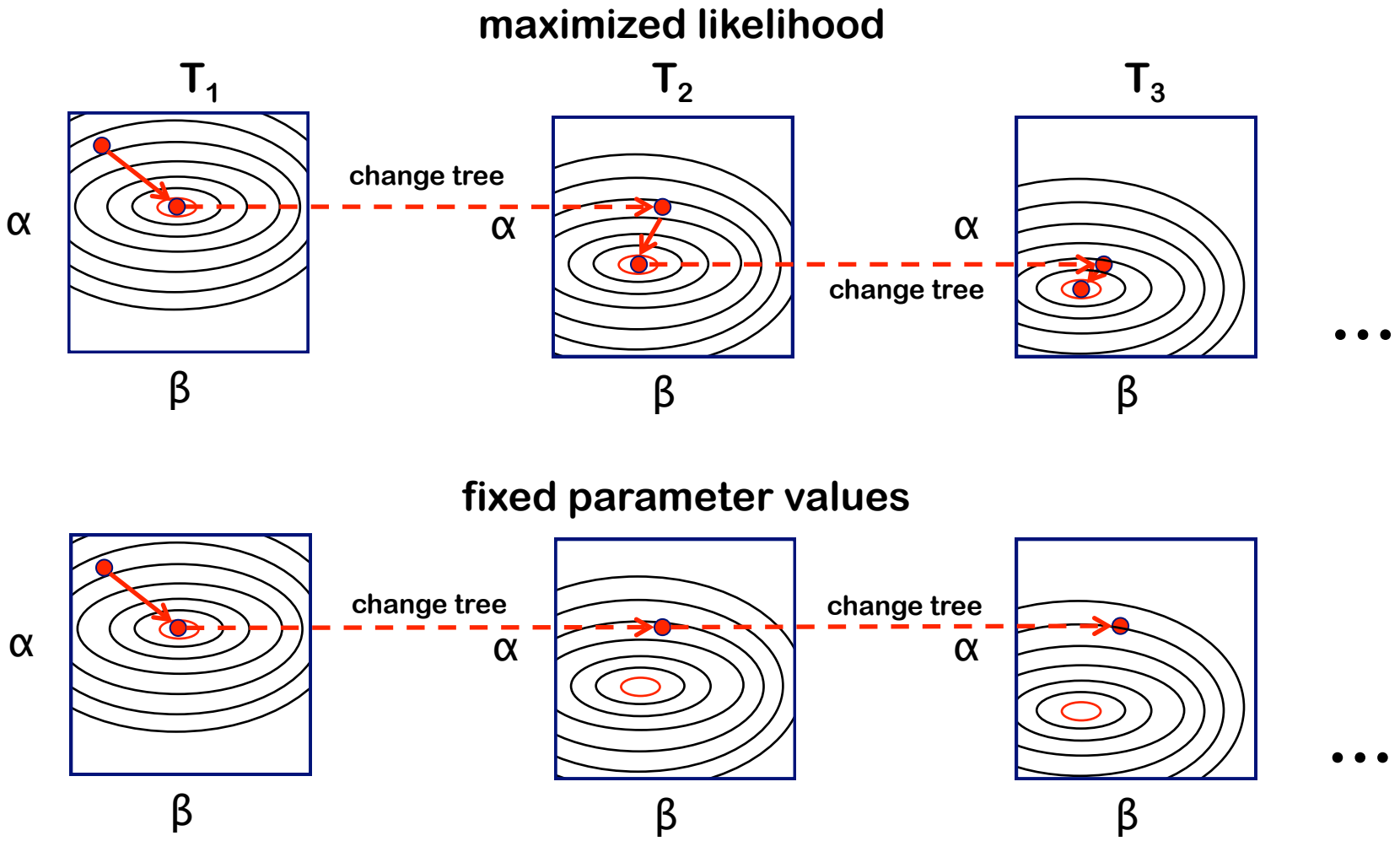
How accurate does a tree likelihood estimate need to be?

(when tree scores are more similar)



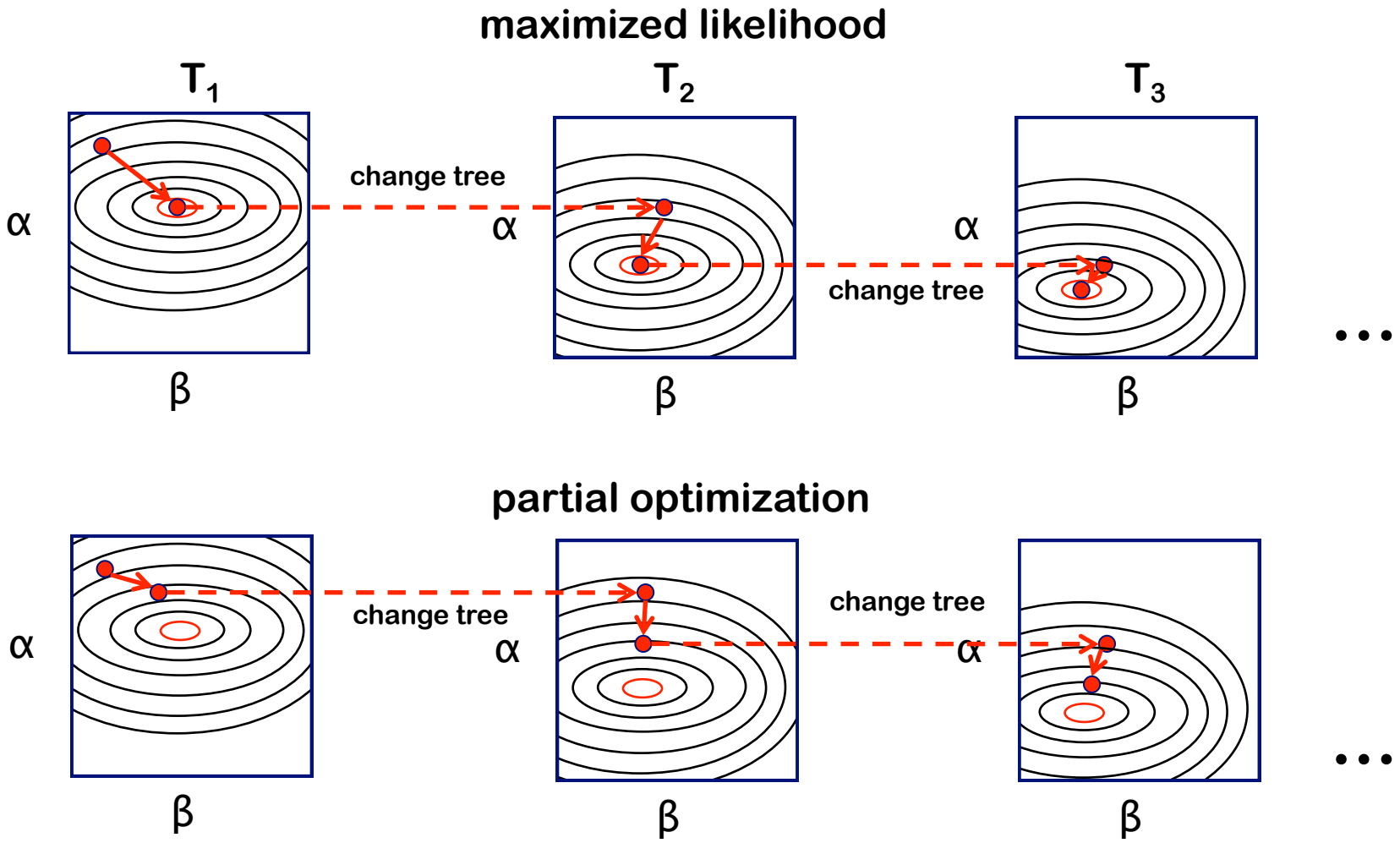
Parameter optima and tree changes

(arbitrary parameters α and β)



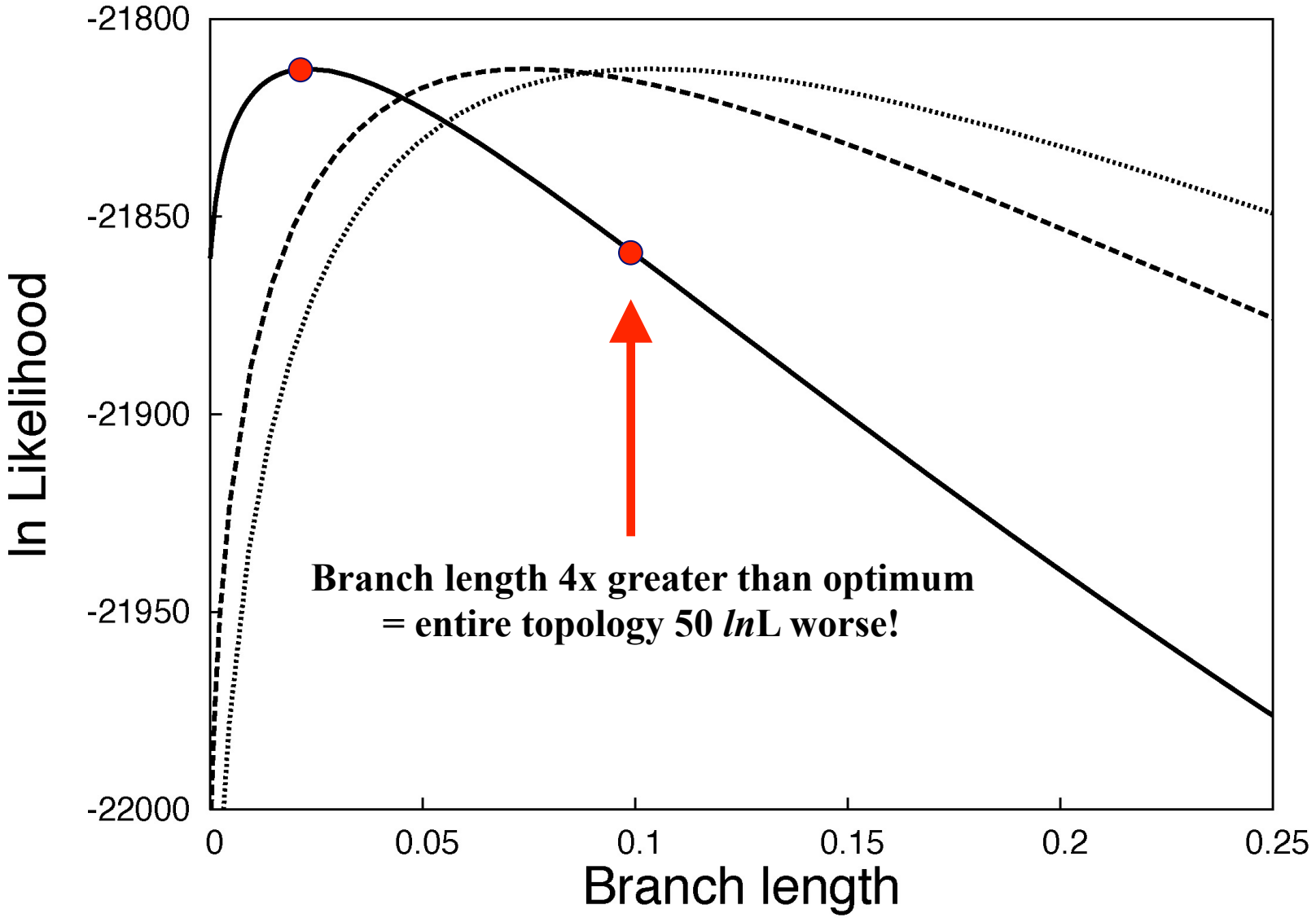
Parameter optima and tree changes

(arbitrary parameters α and β)



How important are branch-length values?

(three example branches in a specific 64-taxon tree)



Branch length importance

If even one branch length is far from optimal, the estimated likelihood will not be useful

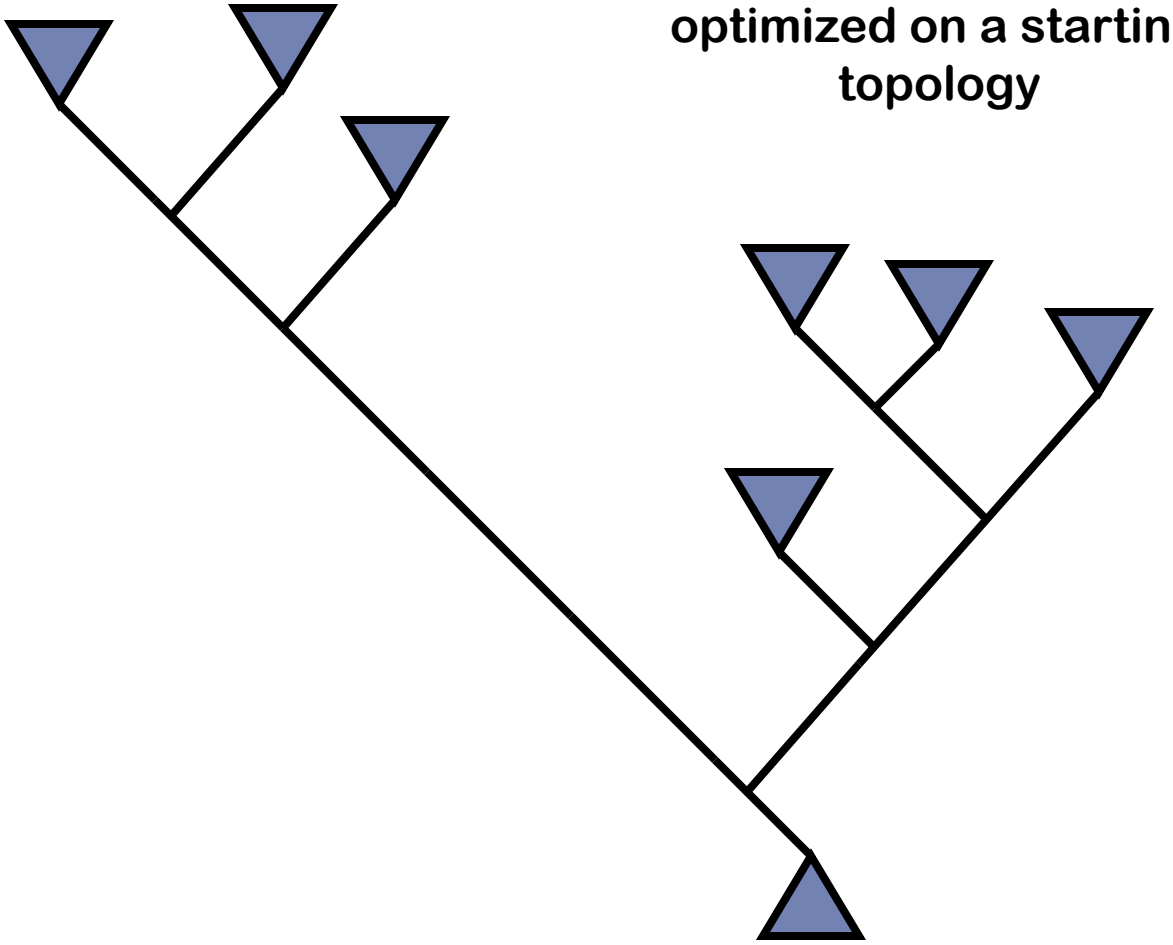
How can we get around optimizing every branch length on every tree?

Using topological similarity

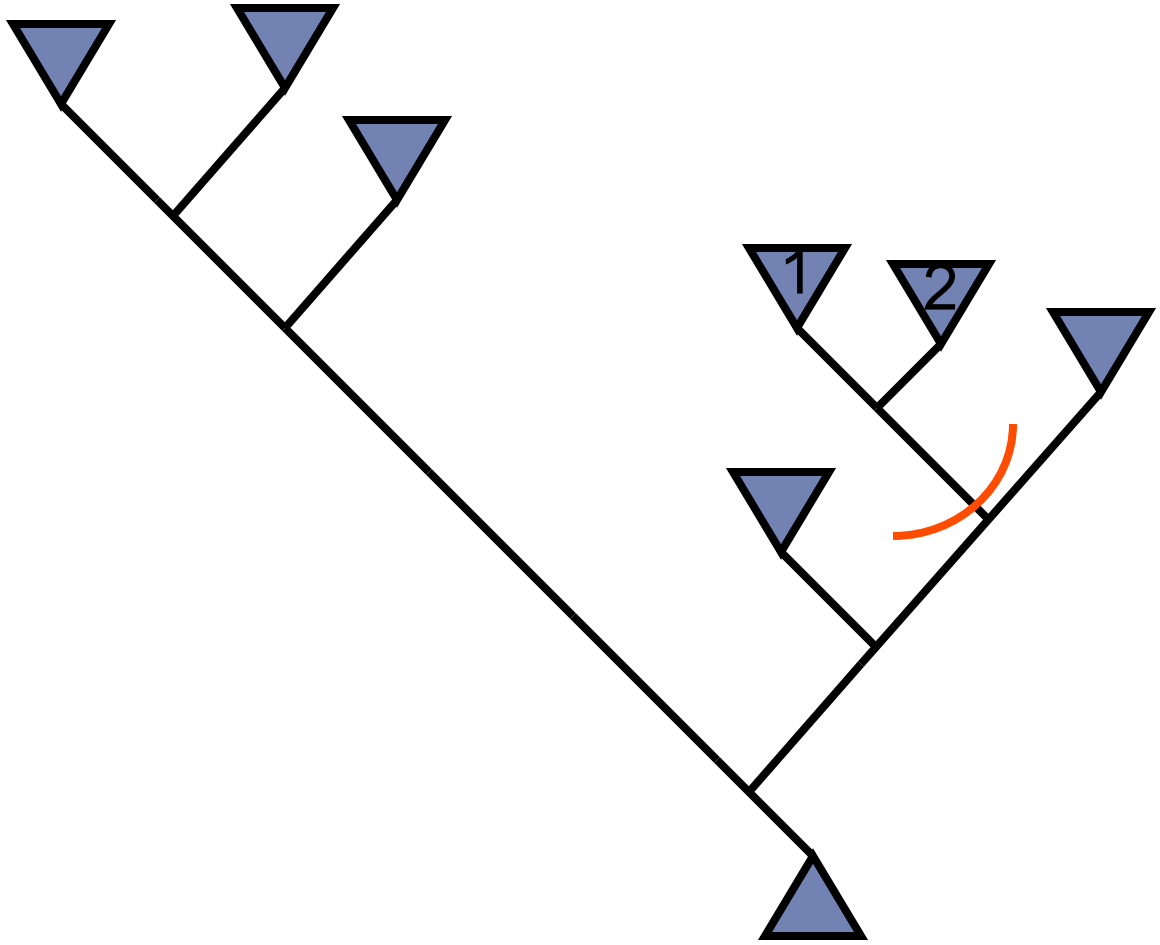
Successive trees are created by slightly modifying an existing tree

We can capitalize on this when dealing with branch-length parameters

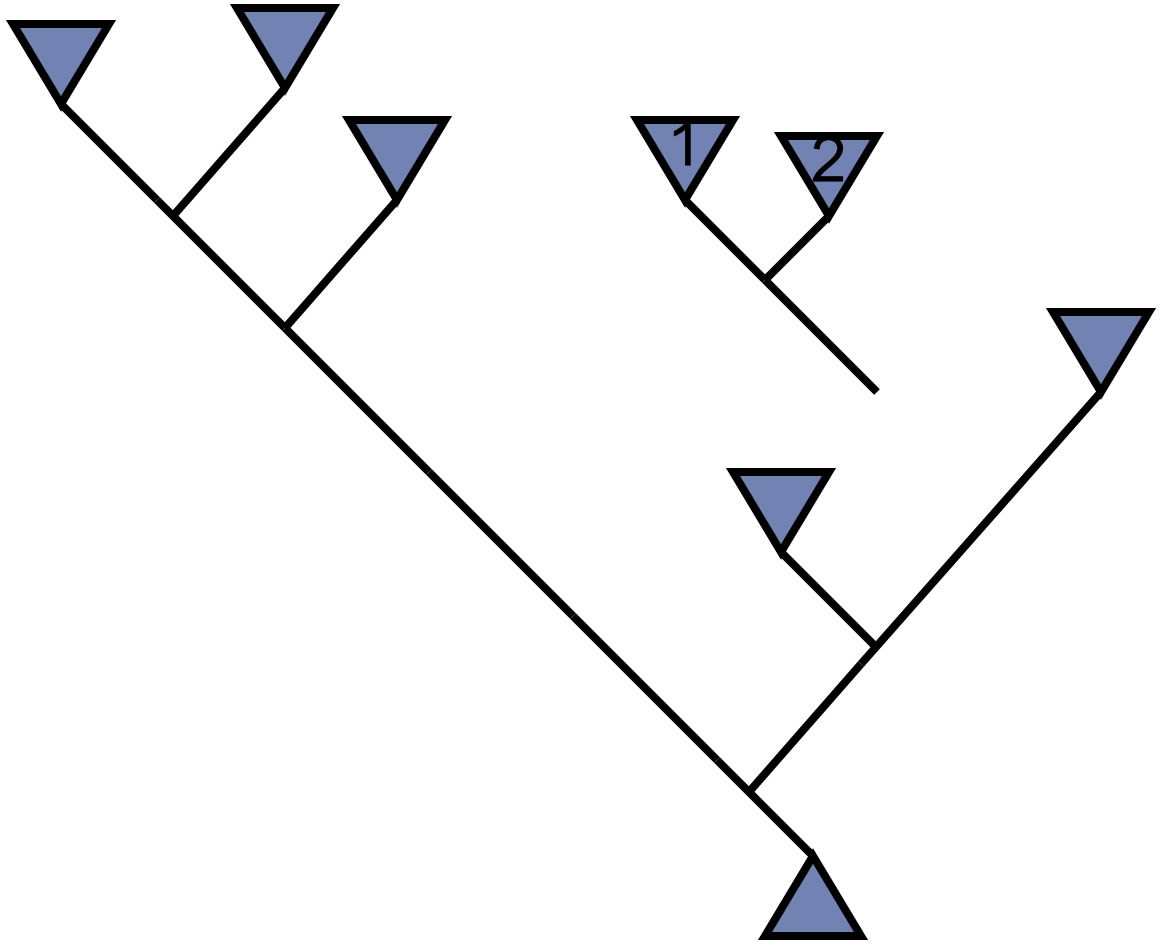
Searching with approximate likelihoods



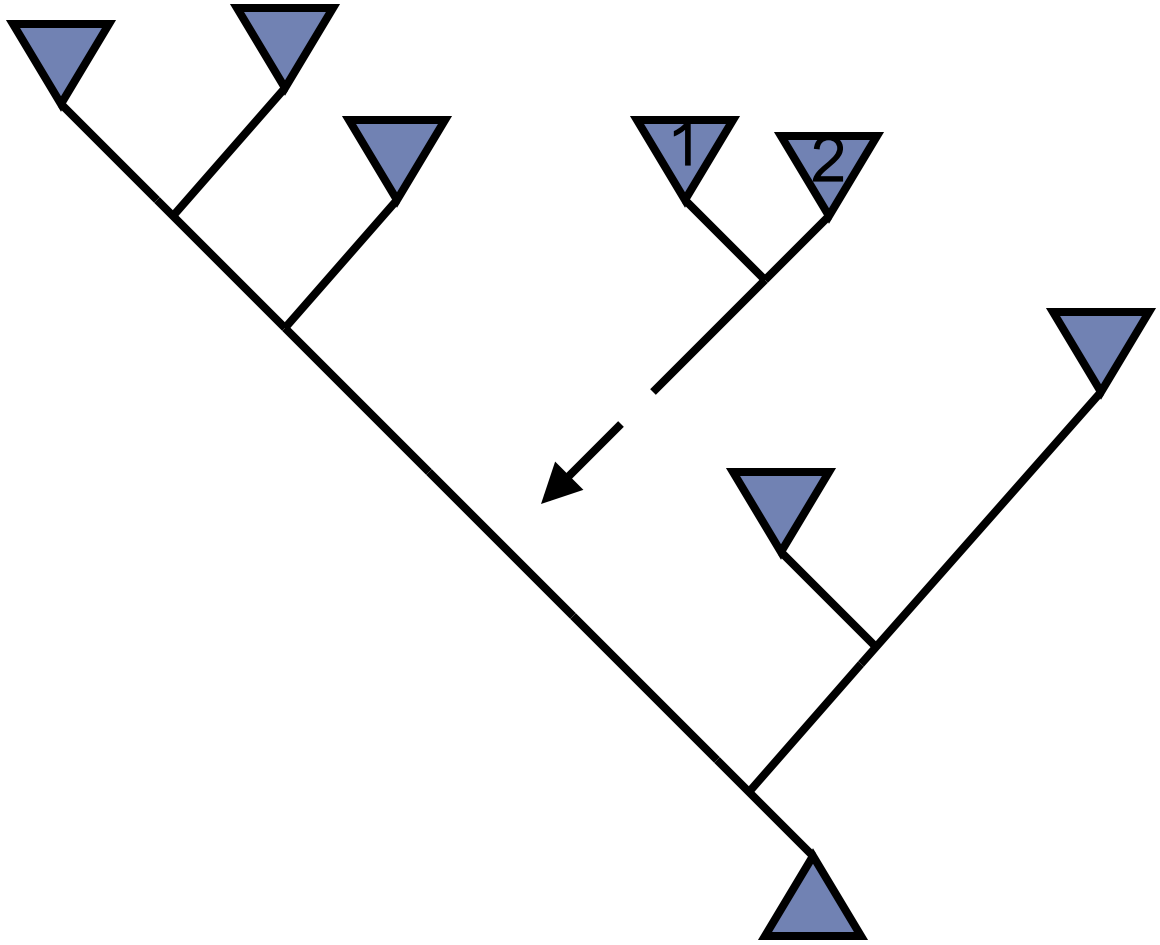
Altering the tree: subtree pruning-regrafting (SPR)



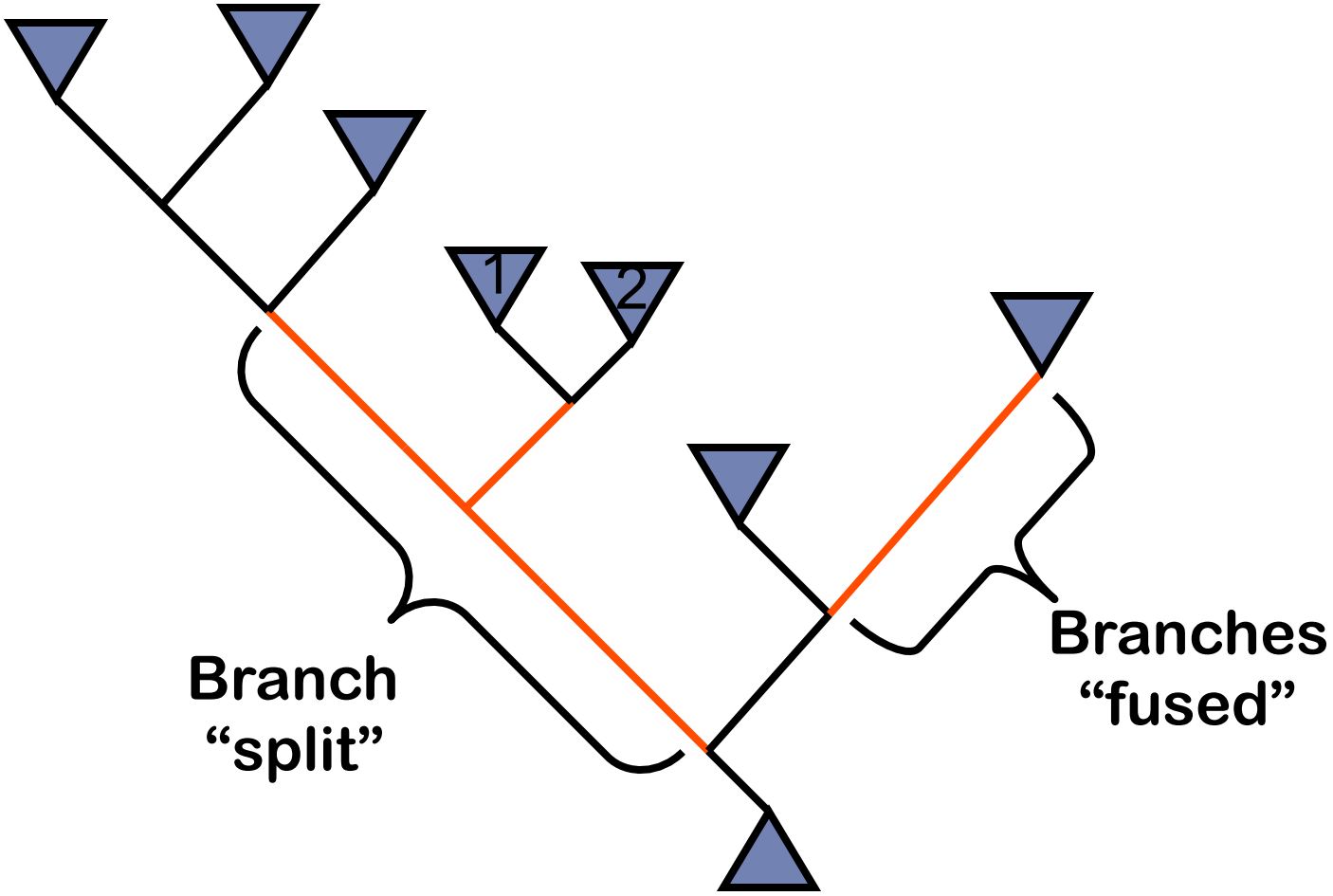
Altering the tree: subtree pruning-regrafting (SPR)



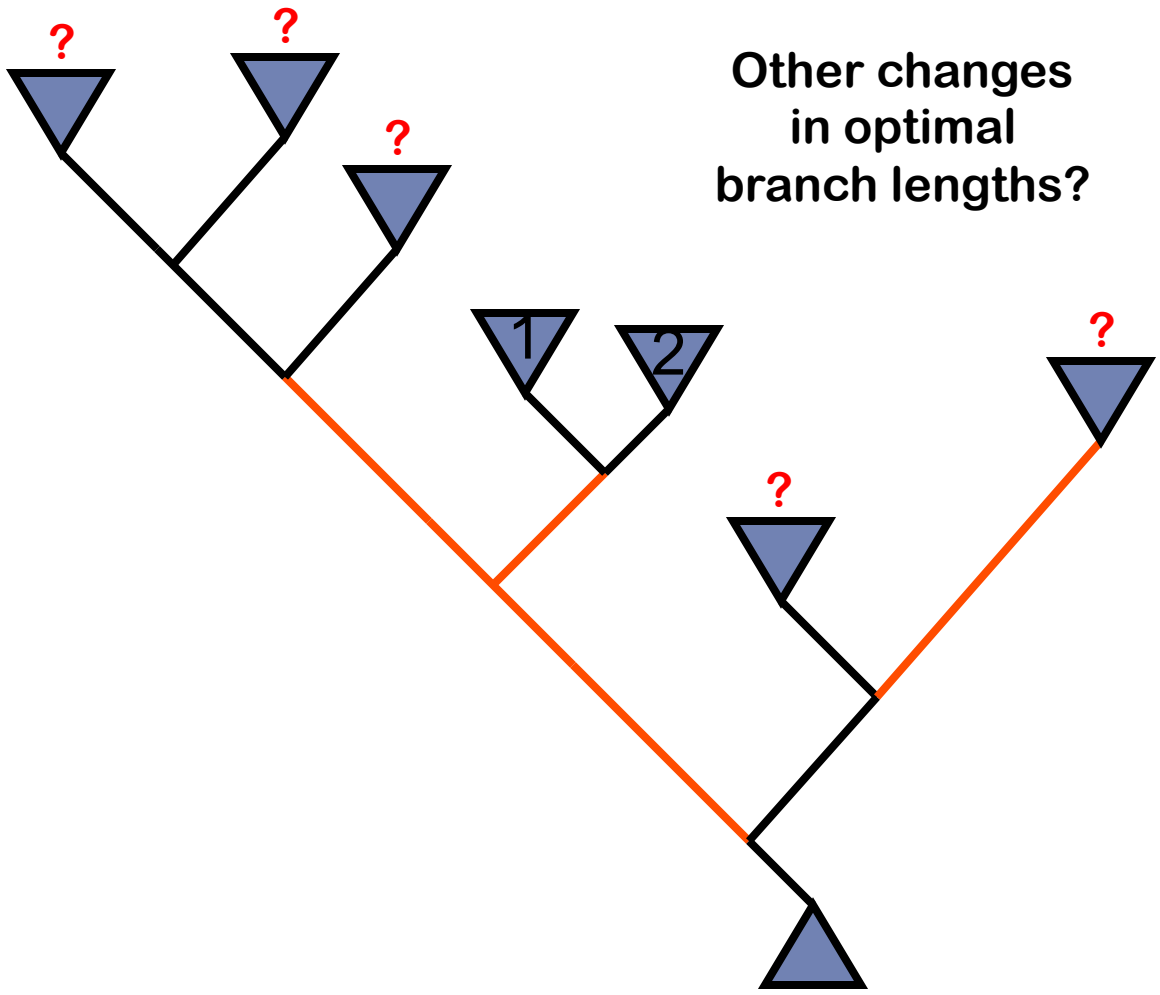
Altering the tree: subtree pruning-regrafting (SPR)



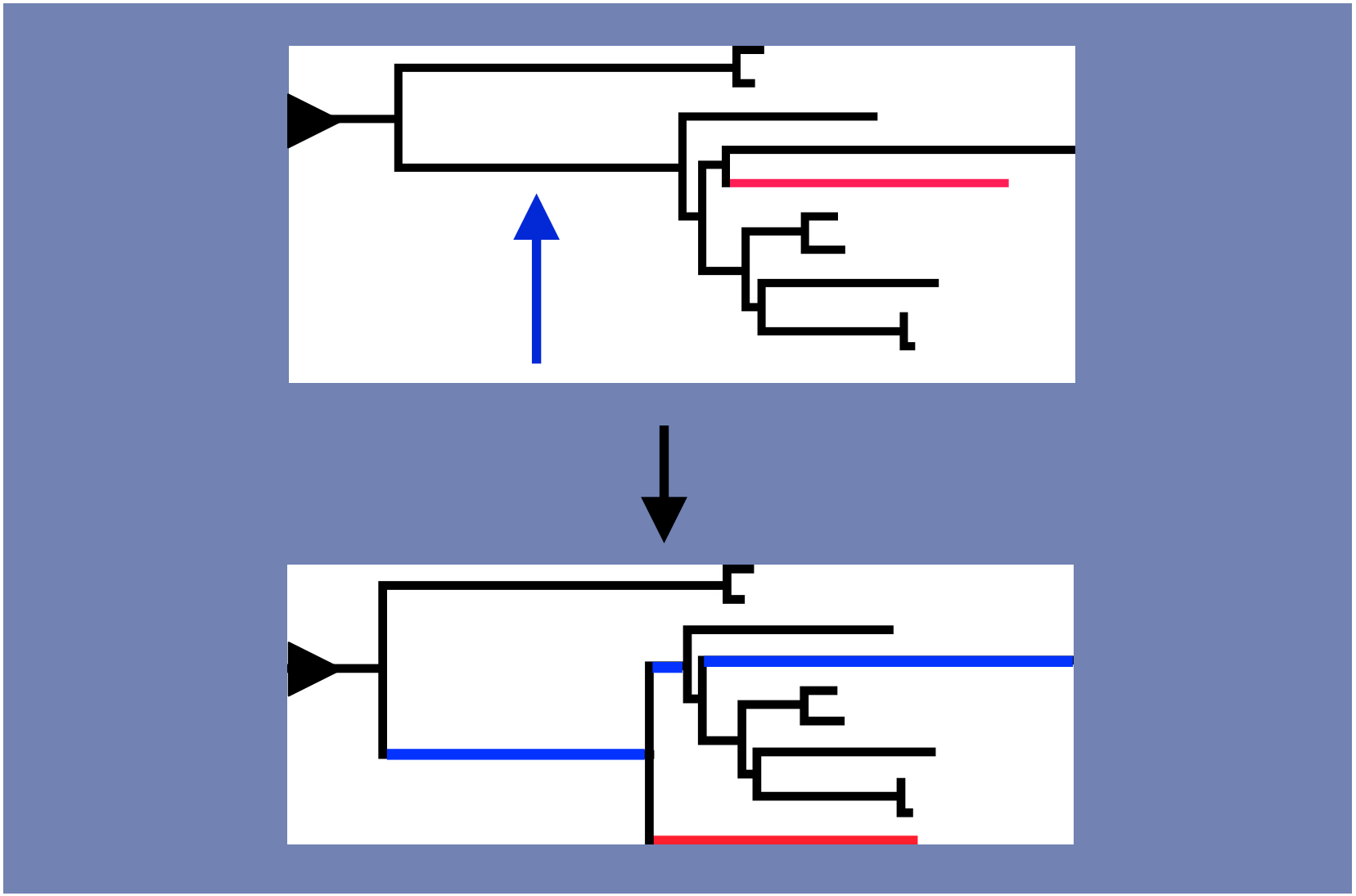
Scoring and optimizing the new topology



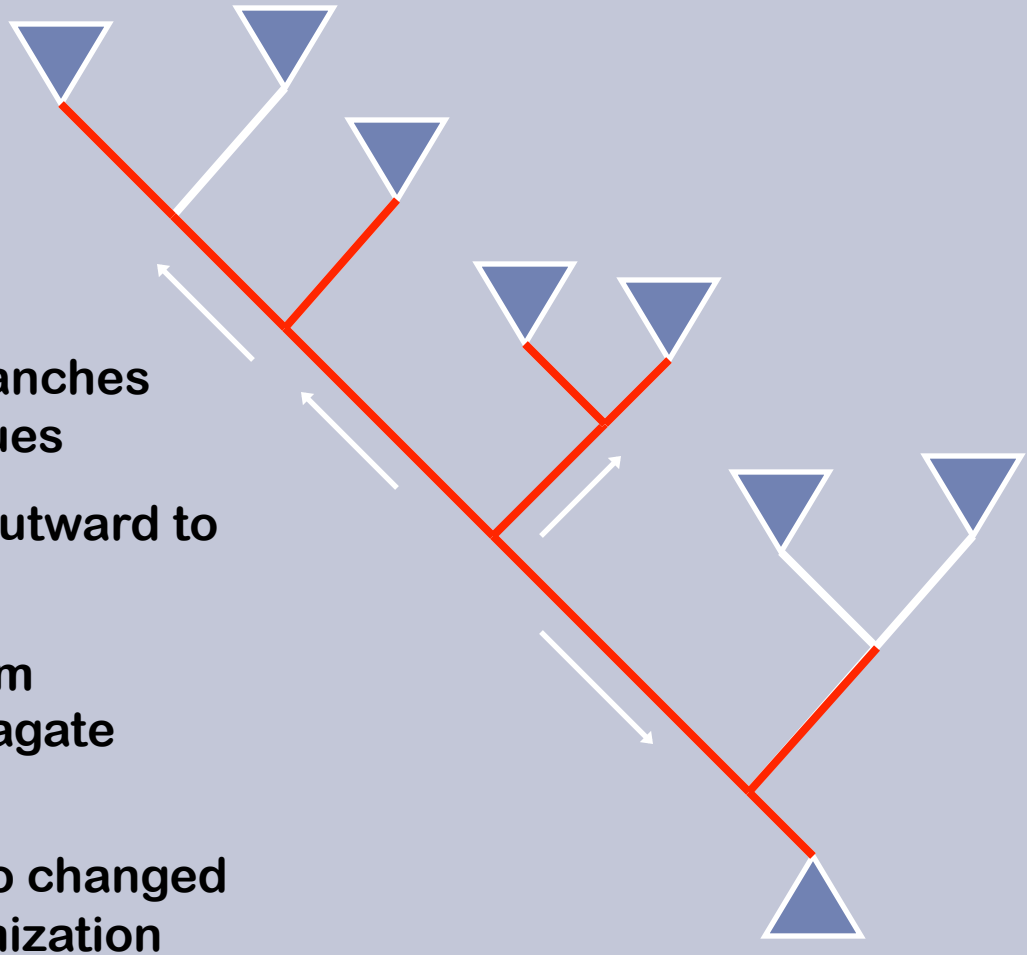
Scoring and optimizing the new topology



Where do optimal branch lengths change?

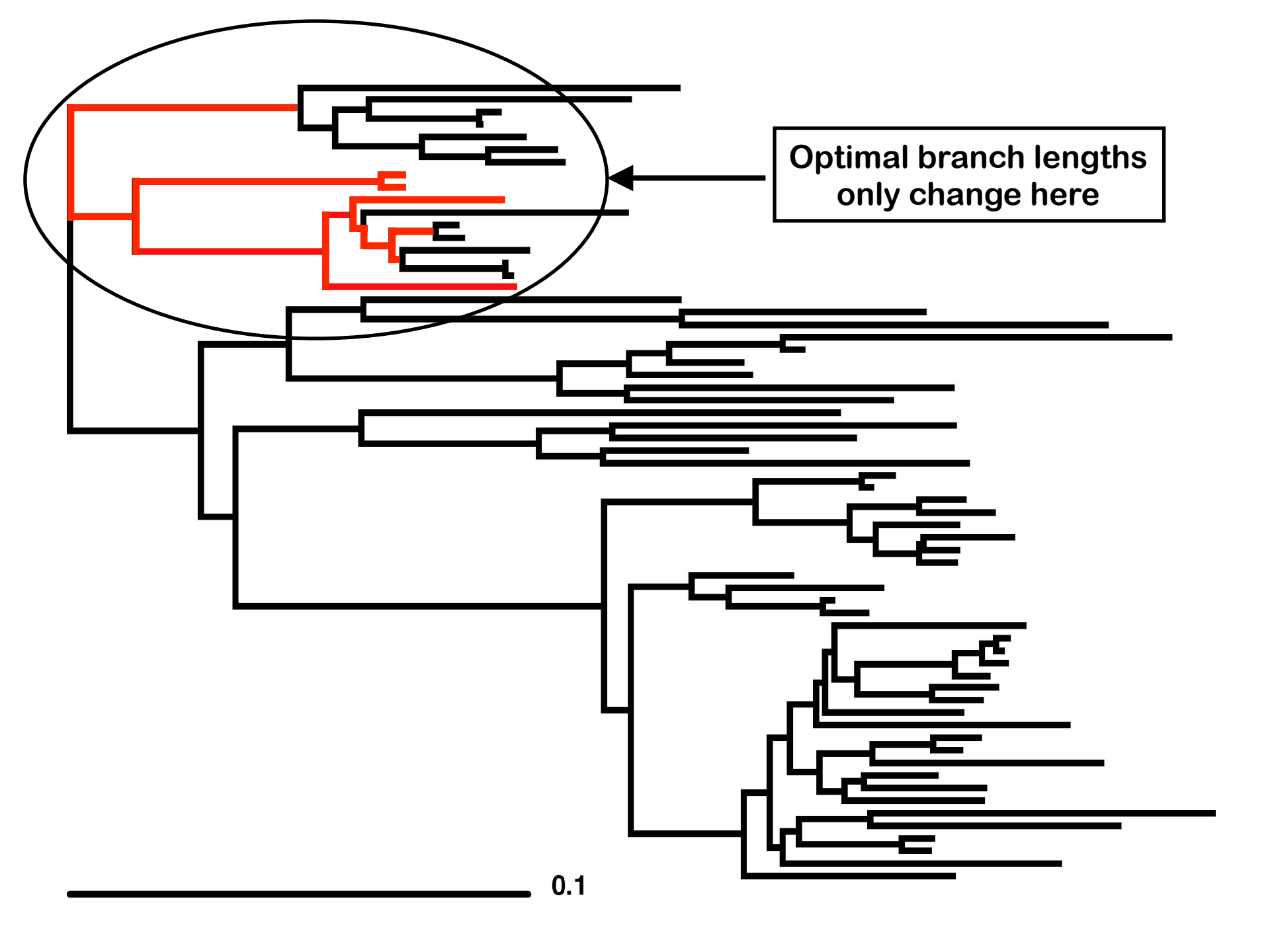


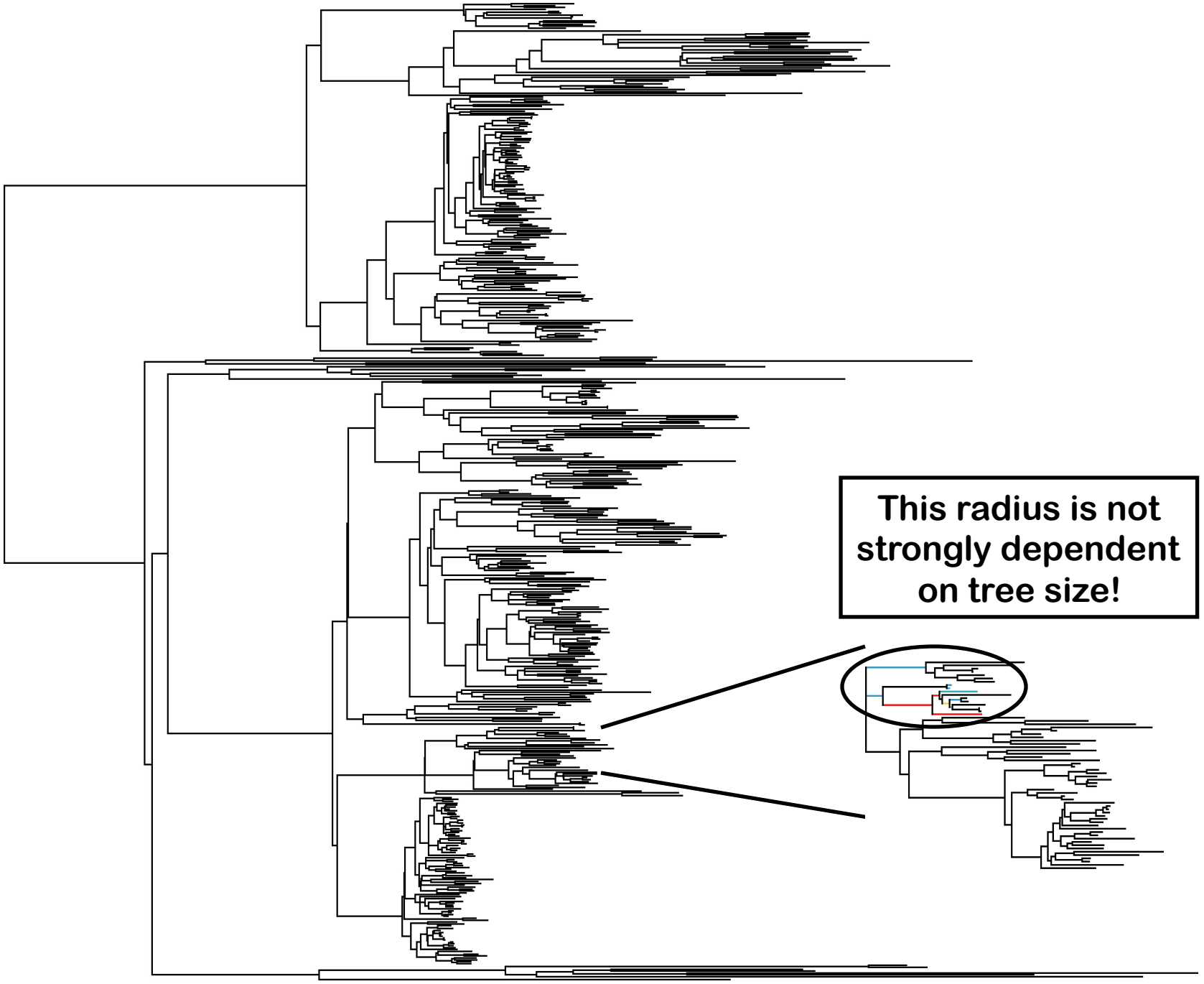
GARLI's post-swap optimization



Optimization rules:

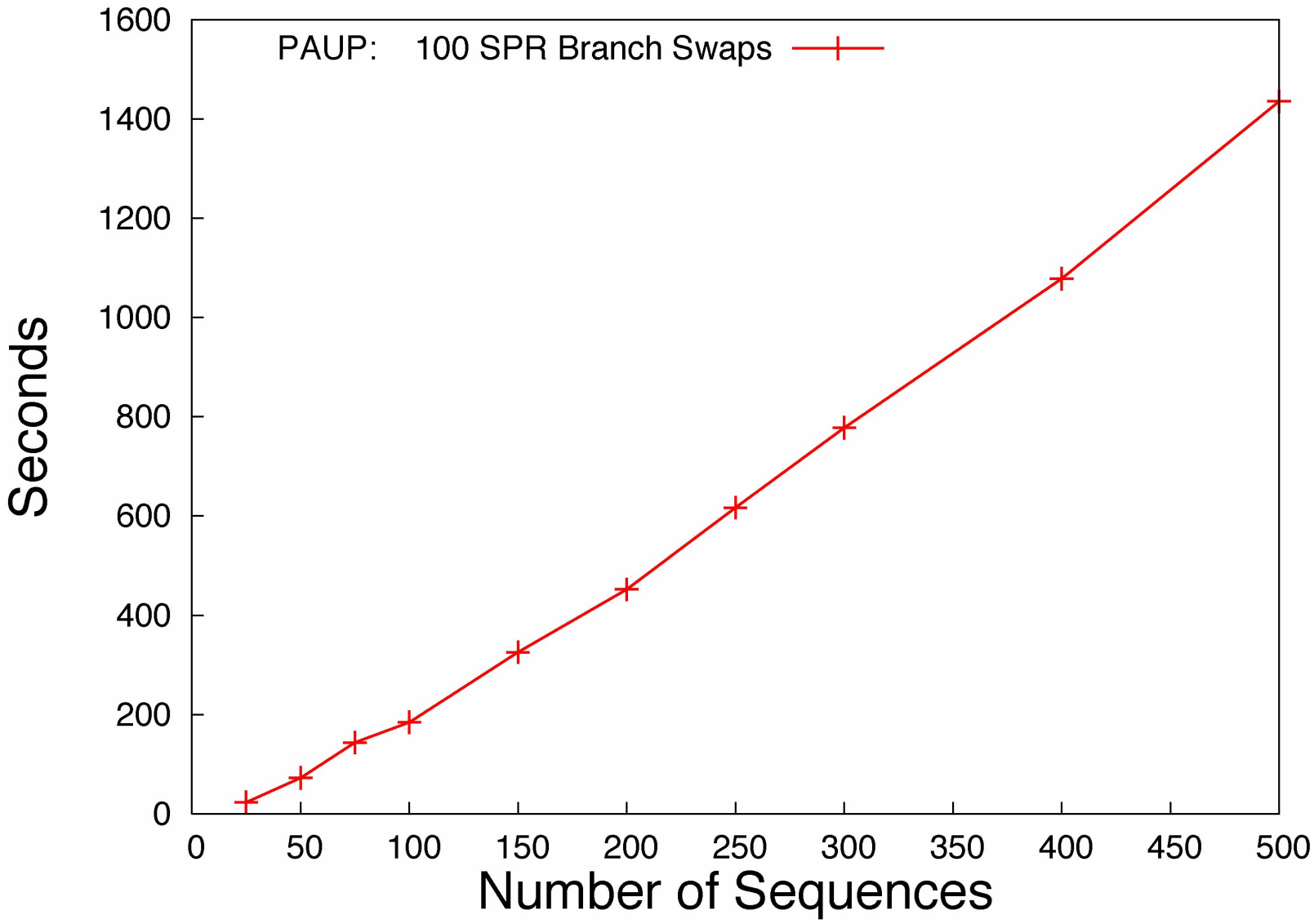
1. Optimize the 3 proximal branches until near their optimal values
2. "Propagate" optimization outward to other branches
3. If a branch length is far from optimum, continue to propagate outward
4. After propagation, return to changed branches for another optimization pass


Optimal branch lengths
only change here
0.1


This radius is not strongly dependent on tree size!

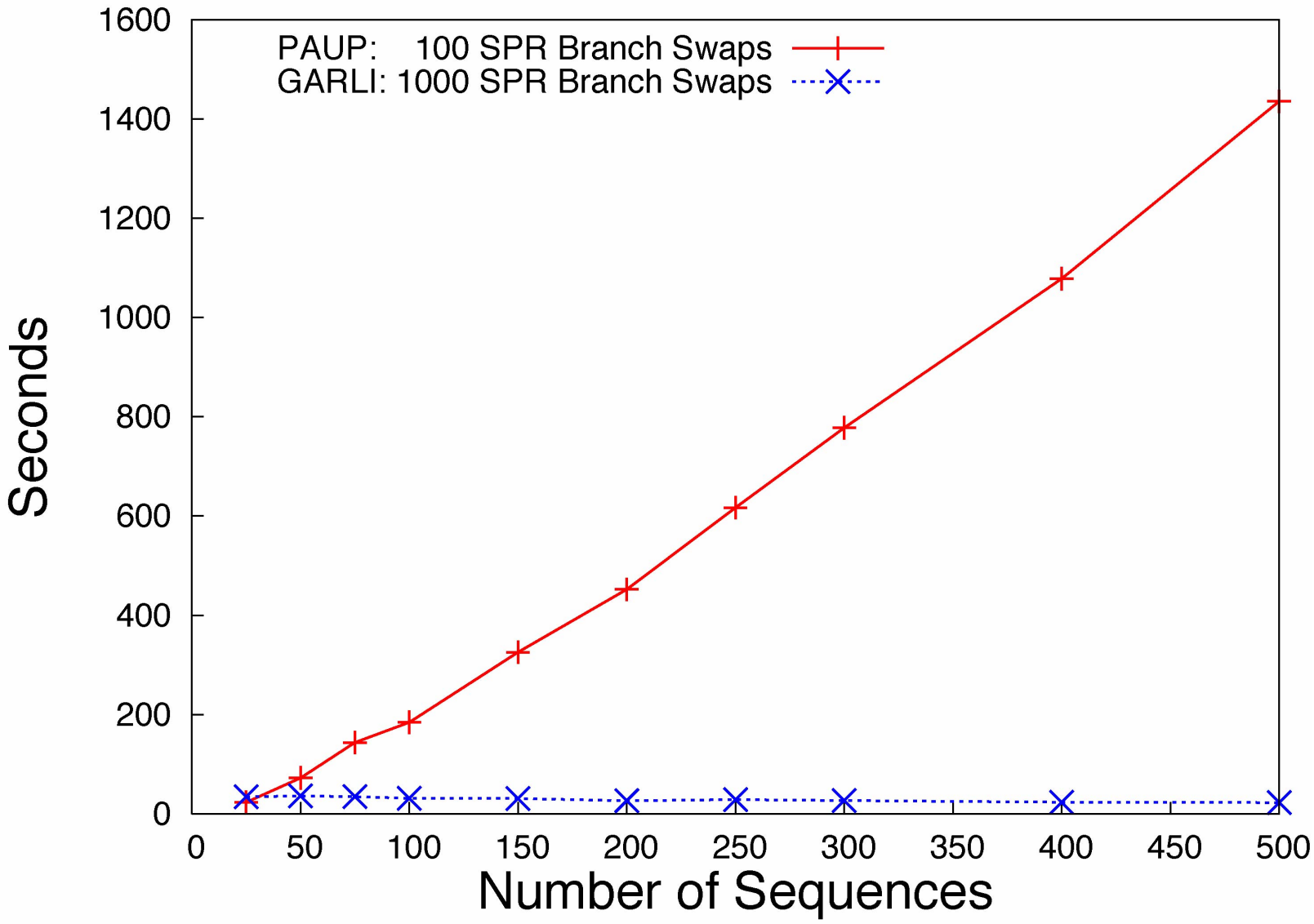
Topology evaluation times

(normalized with respect to # of site patterns)



Topology evaluation times

(normalized with respect to # of site patterns)



Conclusions

GARLI's localized method makes branch-length optimization largely independent of the number of sequences

Several other fast ML heuristics also owe much of their speed to localized optimization (PHYML, RAxML)

What about model parameters?

Model parameters also have different optimal values depending on the topology, branch lengths and other parameter values

Fortunately, strong correlations between optimal model parameter values and topologies are rare

GARLI avoids optimizing model parameters on each tree

Using GARLI in practice

Using GARLI in practice

Performance comparisons (brief)

Allowed models

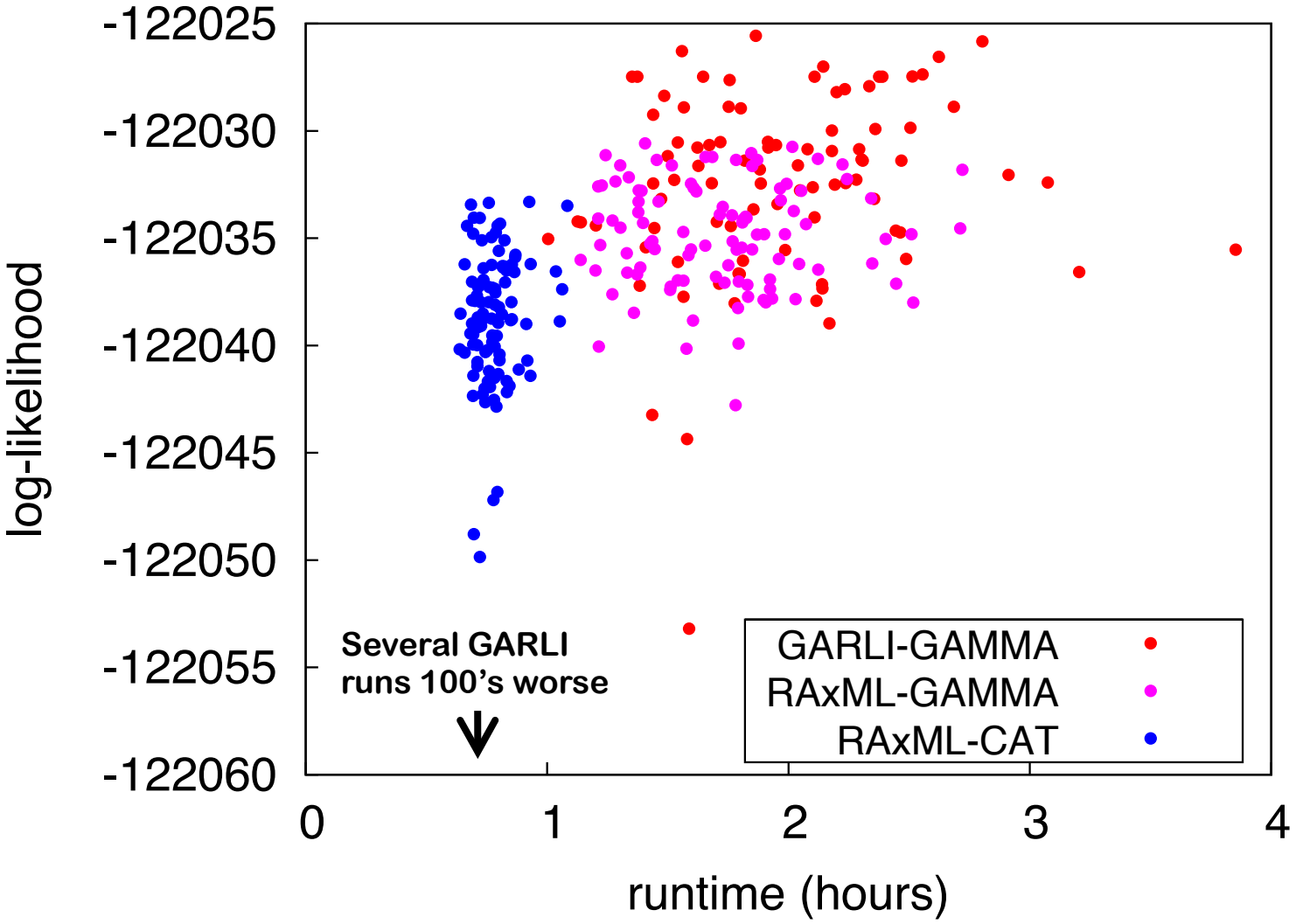
Search strategies

Performance comparisons against other software

More subjective than one would like:

- **What constitutes comparable analyses?**
- **What criteria should be used to compare methods?**
- **Models and likelihood values often not exactly comparable**
- **Most software can be "tuned" to perform better on any particular dataset**
- **Simulated datasets are far too easy to analyze**

Performance comparison:
228 taxon x 4811 nucleotide dataset



ML tree inference software

Some of the most used (alphabetically): GARLI, PAUP*, PHYML, RAxML

For small datasets (< 50 taxa), all of the ML tree inference programs perform well

For large datasets (hundreds of sequences):

- PAUP* is very rigorous, but slowest
- RAxML is generally the fastest
- GARLI often has a slight edge over RAxML in optimality (although often more variability)

RAxML is very efficient for huge datasets (1000+ sequences)

ExaML is the best tool for really huge datasets

ML tree inference software

NOTE: There can be substantial differences in which program performs best depending on the specific dataset!

If the model you need is implemented in multiple software packages, you should use as many of the software tools as you can.

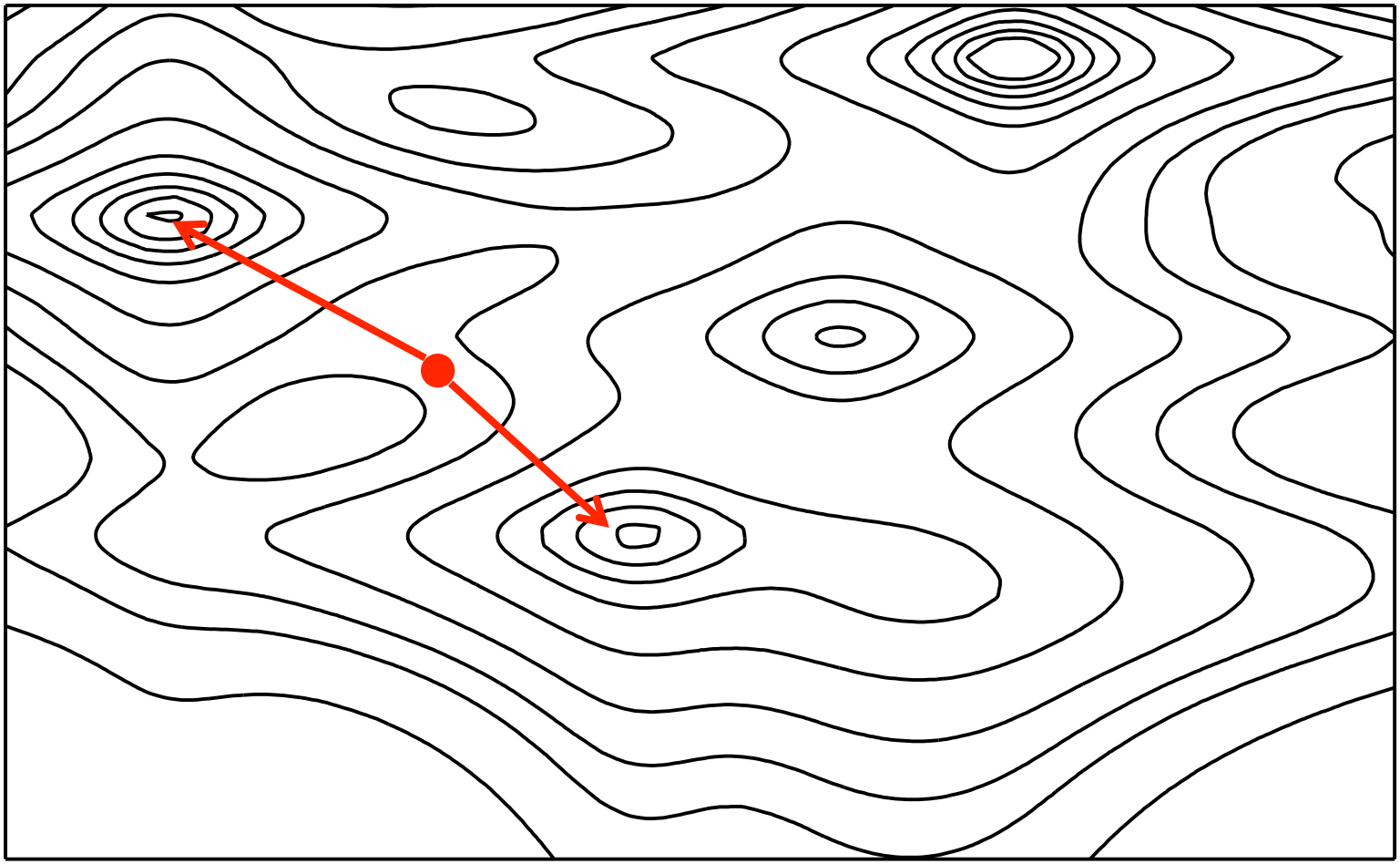
Search strategies in GARLI

Multiple search replicates must ALWAYS be done

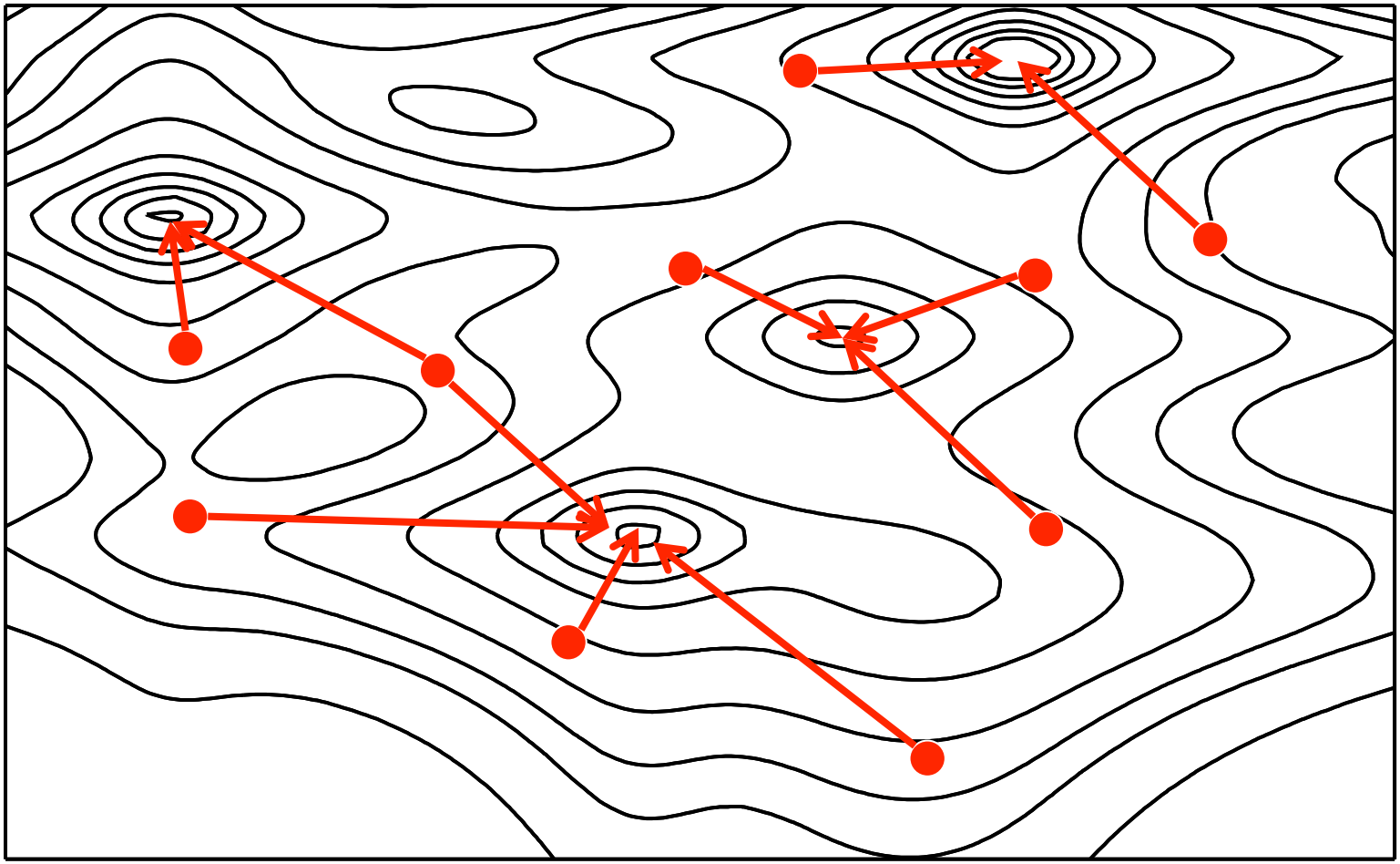
If variable results across search replicates seen:

- **Make changes to improve the search**
- **and/or do more search replicates**

Search repeatability and multiple replicates



Search repeatability and multiple replicates



Search difficulty

On average:

- More sequences = worse
- More characters (signal) = better

parsimony informative sites better indicator of signal than total # of sites

Tuning search intensity

Tradeoff between search intensity and runtimes

Not always a direct relationship between search intensity and solution optimality

Given a certain amount of time, how can we best use it?

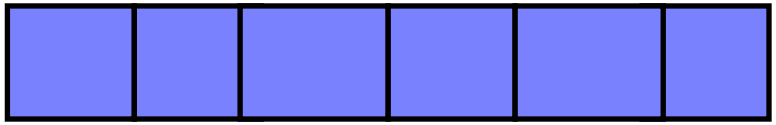
Balancing search intensity and runtimes

H hours

Per run, more likely to find global optimum

3 thorough searches

May be more likely to find global optimum within H hours



6 fast searches

Practical search recommendations

Search repeatability is an indicator of how analyses are going (much like convergence of independent MCMC runs)

Saturating the search space (lots of searches) may be better than very long searches

On some large datasets, unlikely to find the same tree twice

How else can I speed up/improve searches?

Eliminate identical sequences!

Constrained tree searches won't help (in GARLI)

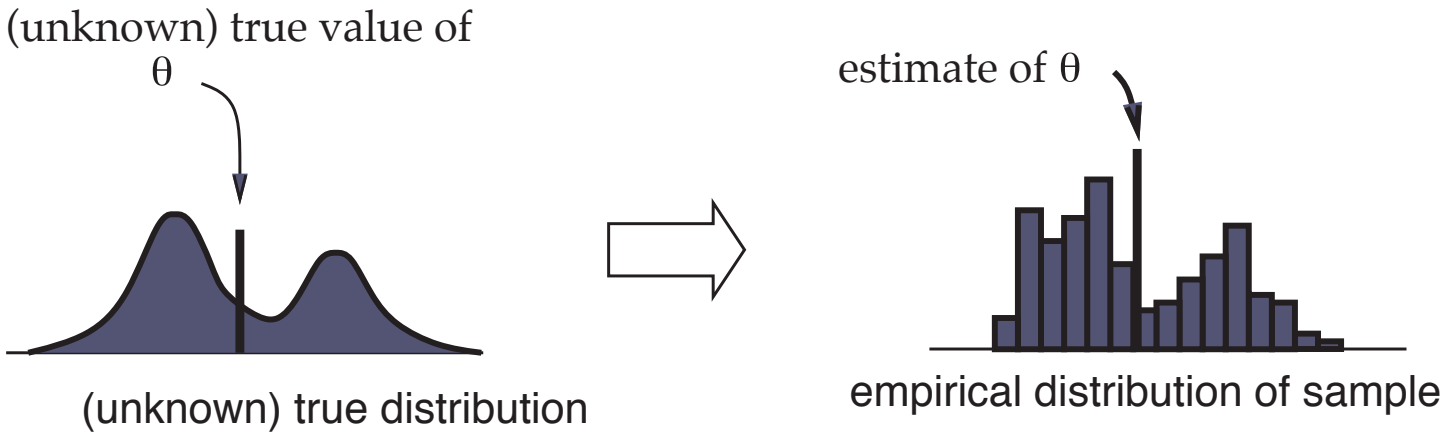
Starting tree

- **Providing a decent (potentially unresolved) starting tree can help on large datasets**

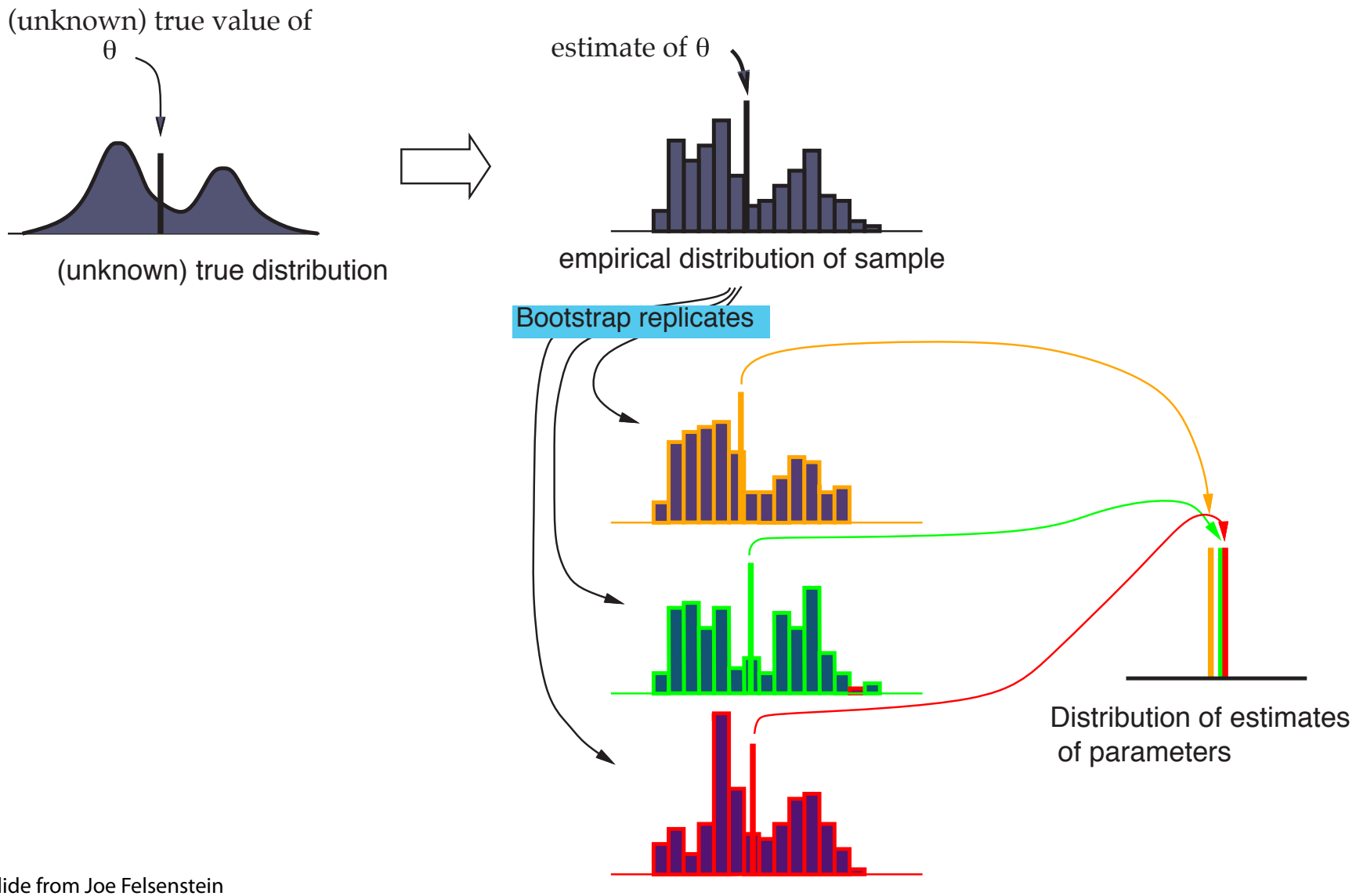
Bootstrapping

- The ML tree is just a point estimate
- How much confidence should we have that the groups recovered are not just an artifact of having a small sampling of characters?

The bootstrap

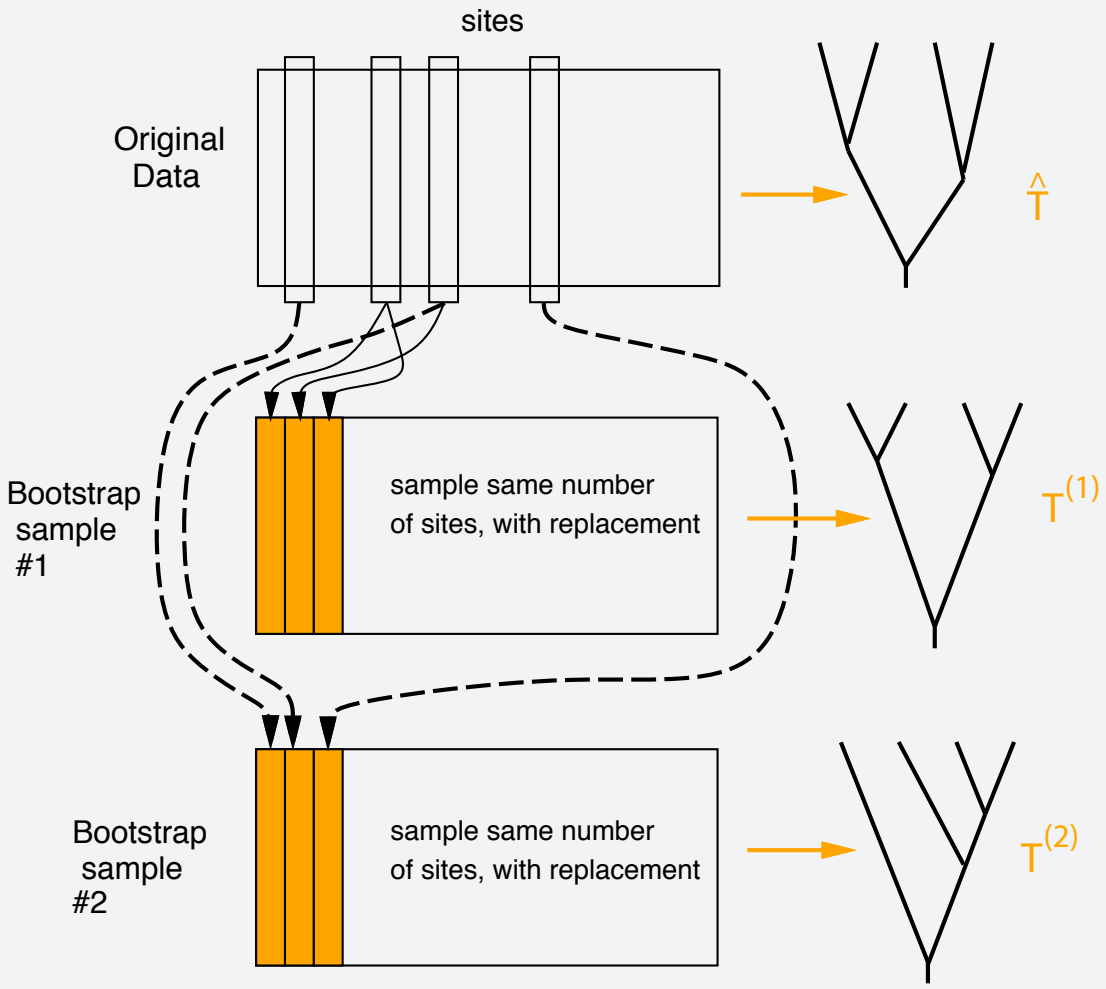


The bootstrap



Slide from Joe Felsenstein

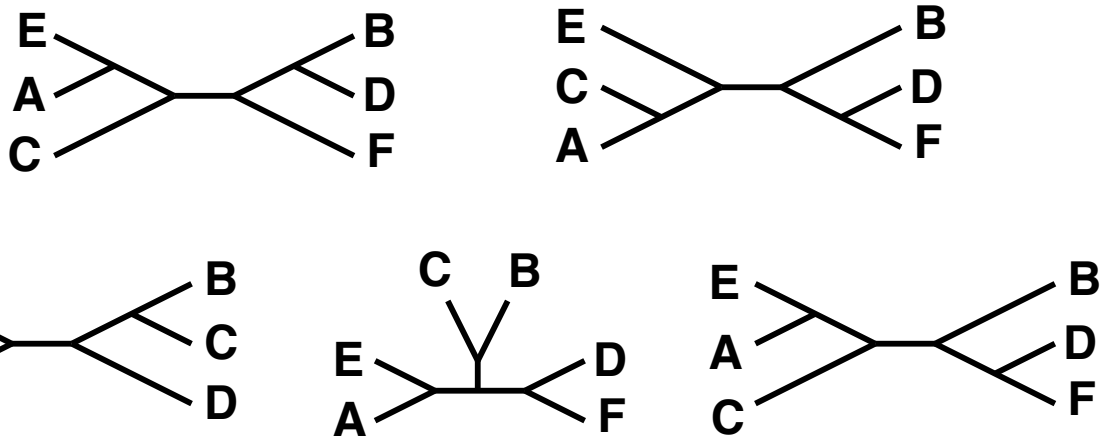
The bootstrap for phylogenies



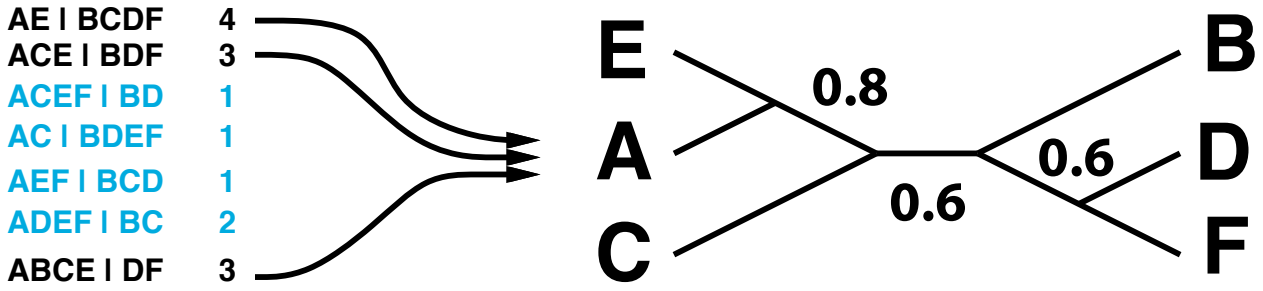
Slide from Joe Felsenstein

The majority-rule consensus tree

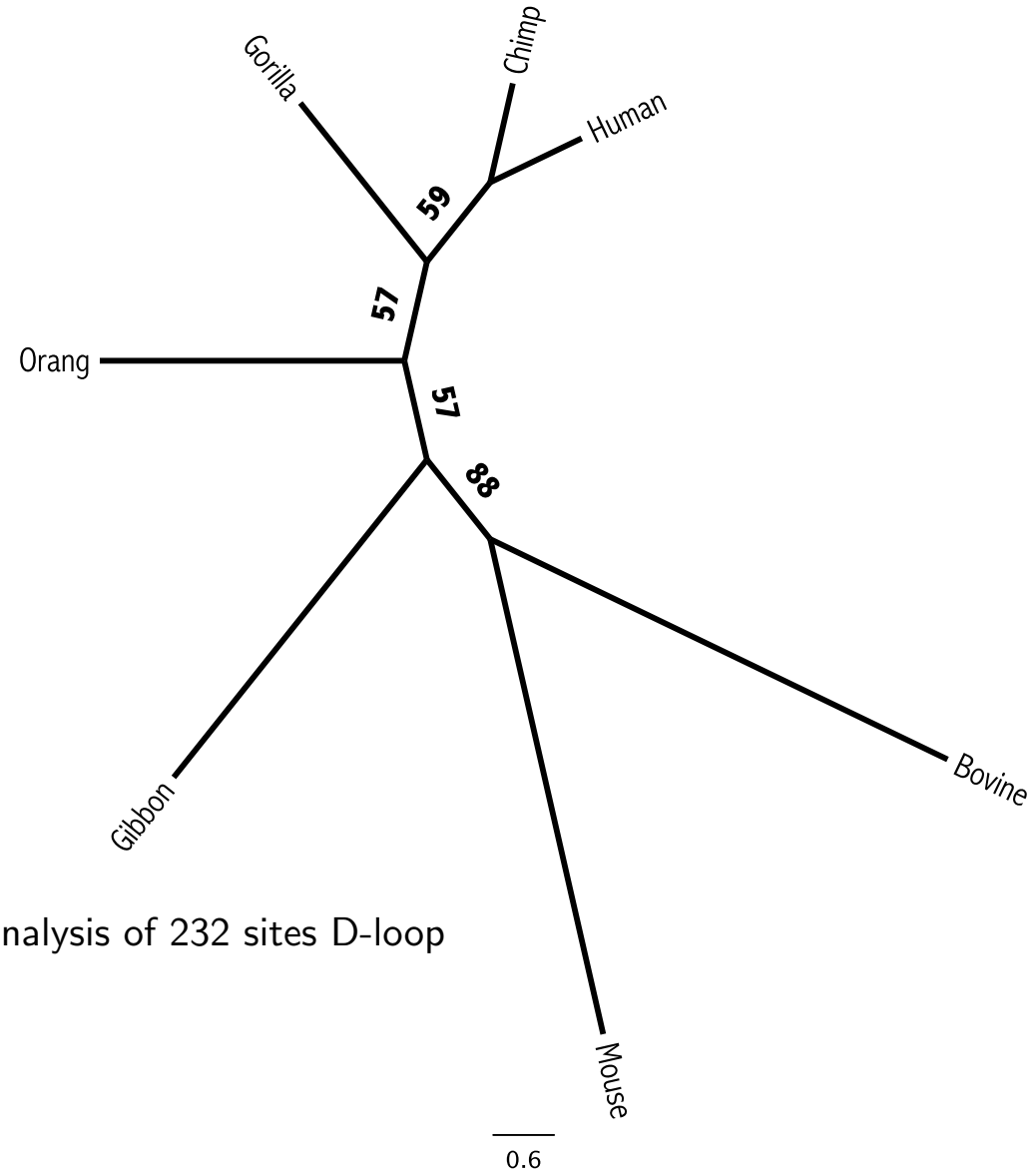
Trees:



How many times each partition of species is found:



Slide from Joe Felsenstein



From Hasegawa's analysis of 232 sites D-loop

Bootstrapping cartoons

The mechanics of bootstrapping:

- http://phylo.bio.ku.edu/mephytis/boot-sample.html

The effect of sample size on bootstrap support:

- http://phylo.bio.ku.edu/mephytis/bootstrap.html

Bootstrapping for branch support

- Typically a few hundred bootstrap, pseudoreplicate datasets are produced.
- Less thorough searching is faster, but will usually artificially lower bootstrap proportions (BP). However, Anisimova et al. (2011) report that RAxML's rapid bootstrap algorithm may inflate BP.
- "Rogue" taxa can lower support for many splits – you do not have to use the majority-rule consensus tree to summarize bootstrap confidence statements.

Bootstrapping in GARLI?

GARLI can run multiple search replicates per bootstrap reweighting, with the best scoring tree saved

More intense searches add up quickly when bootstrapping

Find fastest settings that give consistent results on full data, use those for bootstrap searches

Evolutionary models

GARLI is geared toward model flexibility and rigorous parameter estimation

Model types

- **Any GTR submodel for nucleotides**
- **Various common amino acid models**
- **Simple codon models**
- **Non-sequence data (**$\mathrm{M}k$ **and** $\mathrm{M}k_v$**)**
- **Partitioned models**
- **Indel models (unreleased)**

How/when to partition

PartitionFinder may prove to be a great approach to partitioned model selection

Smaller subsets increase sampling error, lead to parameter estimation difficulties and model breakdown

Over-partitioning may have serious consequences in ML inference, less in Bayesian

Non-bifurcating trees

GARLI returns trees with polytomies when branches have an optimal length of zero, but some programs do not.

This can become very important in low divergence phylogenomic studies

Assorted GARLI features

Single data file may be analyzed at the nucleotide, amino acid and codon levels without making changes to it.

Multithreaded version for multiple CPU cores

MPI version simplifies bootstrapping on clusters.

Full checkpointing

Topological constraints (positive, negative, backbone)

Other assorted GARLI features

Specification and fixation of model parameter values

Site-likelihood output for all models including partitioned, for input into CONSEL, etc.

Ancestral state reconstruction for all models

Eventually: Beagle GPU version

Gap (indel) models

Using gap characters

GARLI implements two models appropriate for use on gap characters in *fixed alignments*

DIMM model based on Rivas and Eddy (2008)

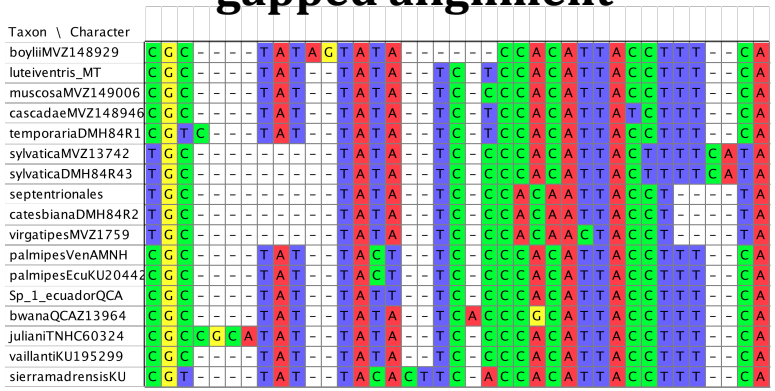
- **Real model of insertion-deletion process**
- **Non-reversible, dollo (one insert, multiple delete)**

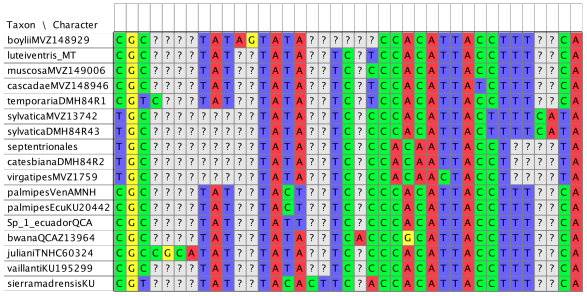
Variant of $\mathrm{M}k_v$ (Lewis, 2001) model (no full gap columns)

- **Allows many gap-base transitions**

Fixed alignment analysis options

gapped alignment

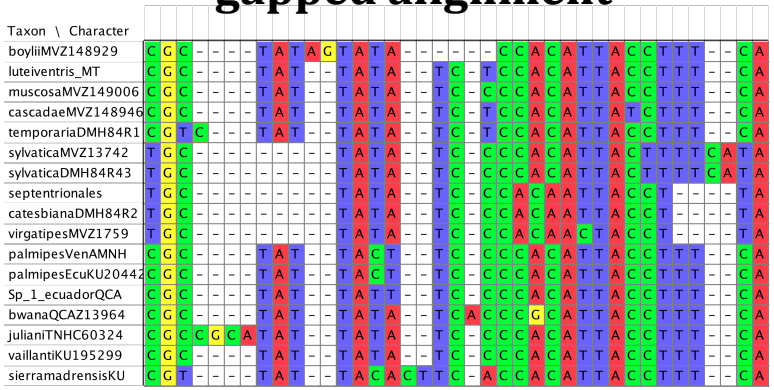


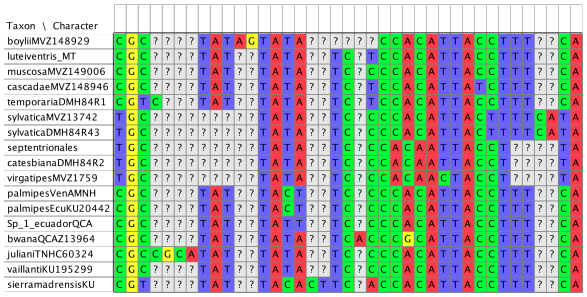


ignore gaps, e.g. GTR (the usual)

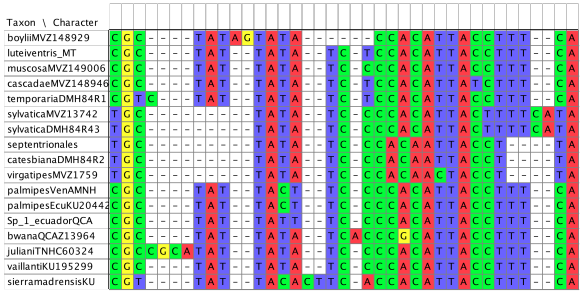
Fixed alignment analysis options

gapped alignment





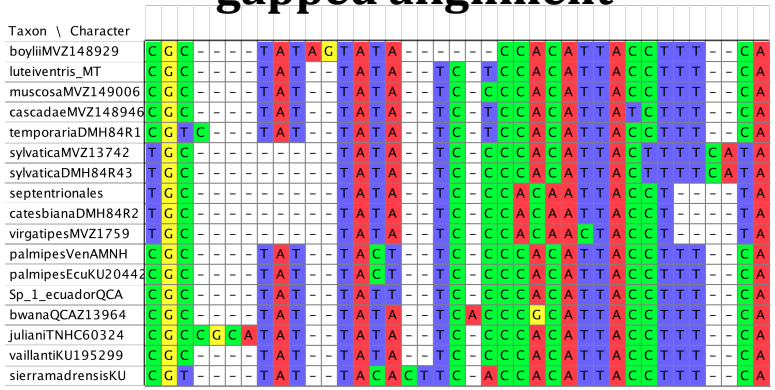
ignore gaps, e.g. GTR (the usual)



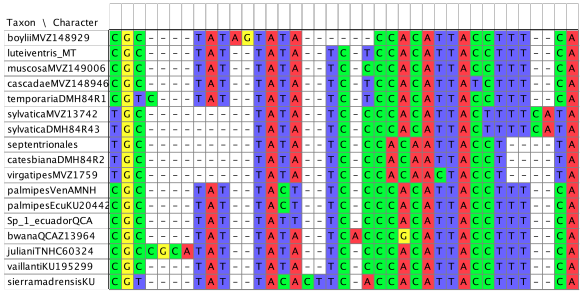
5-state approach
(e.g., Rivas and Eddy)

Fixed alignment analysis options

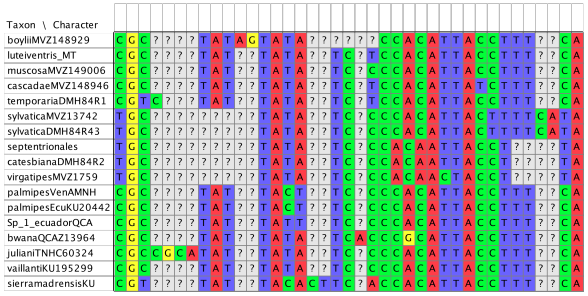
gapped alignment



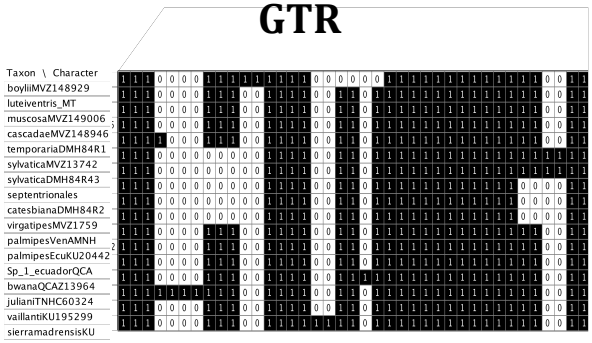
5-state approach
(e.g., Rivas and Eddy 2008)



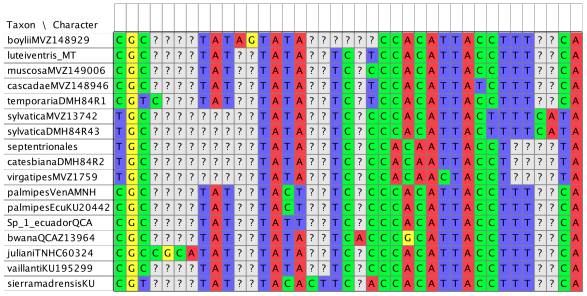
“gapcode” matrix & use partitioned model



GTR

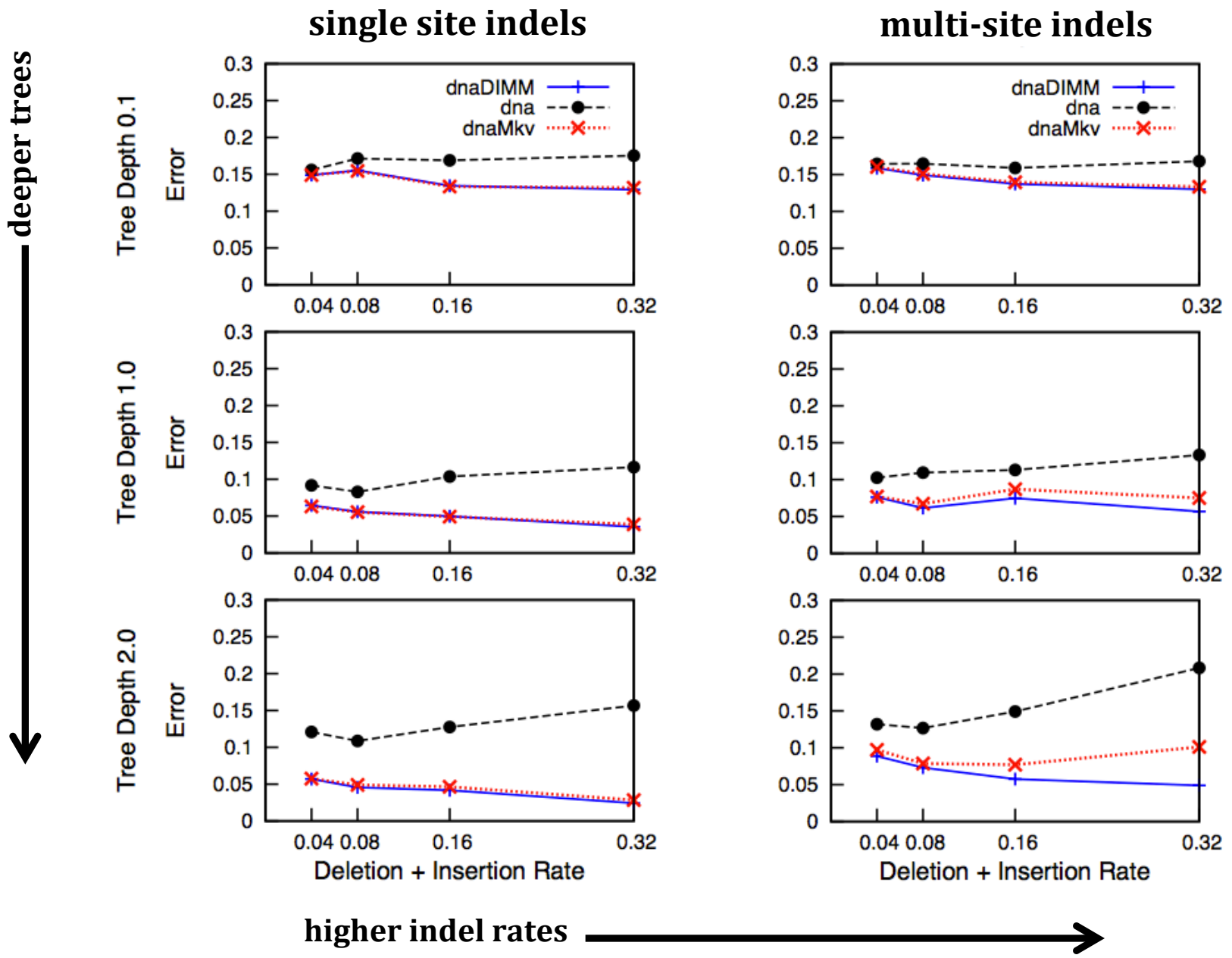


DIMM
(or Mk_v)

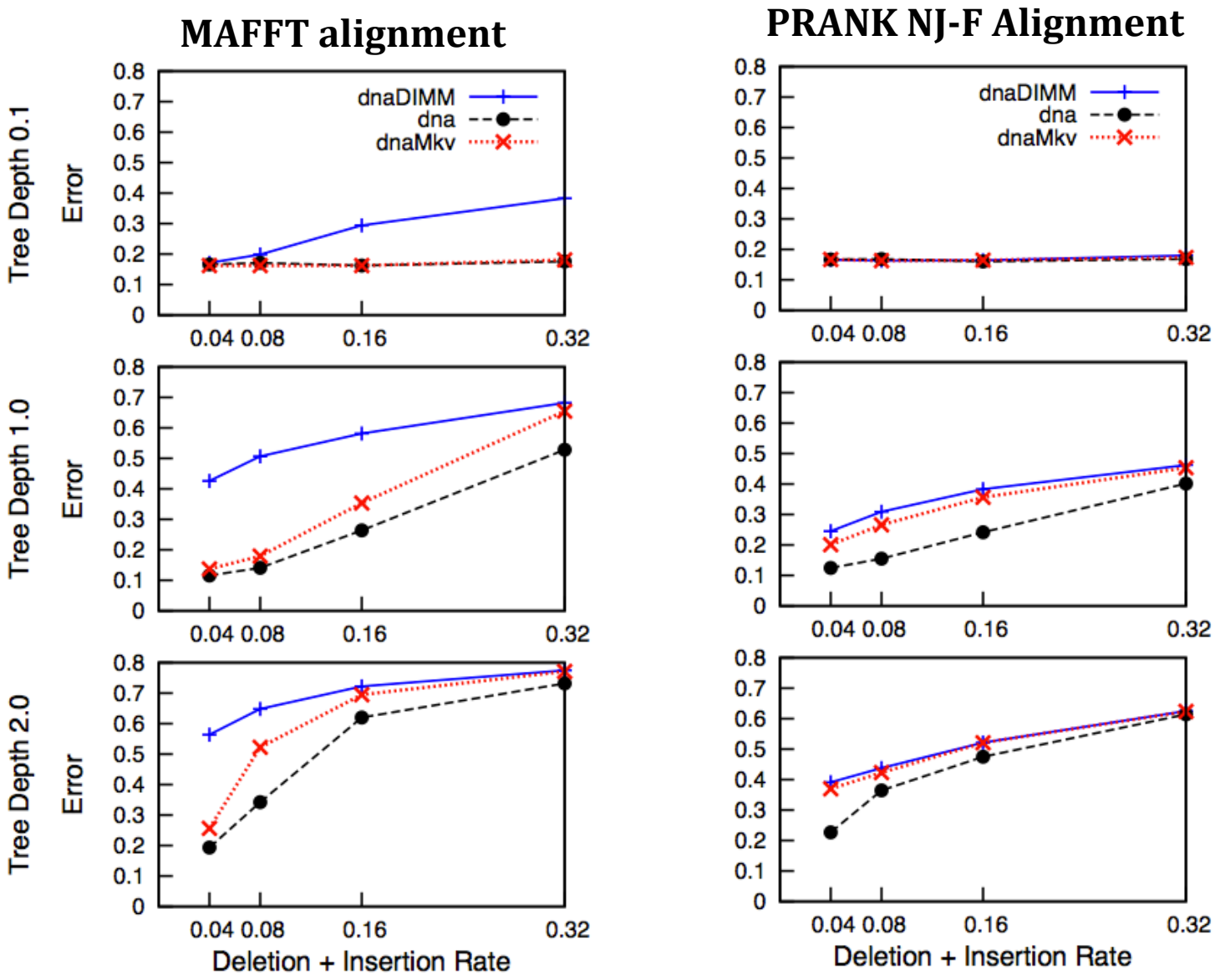


ignore gaps, e.g. GTR
(the usual)

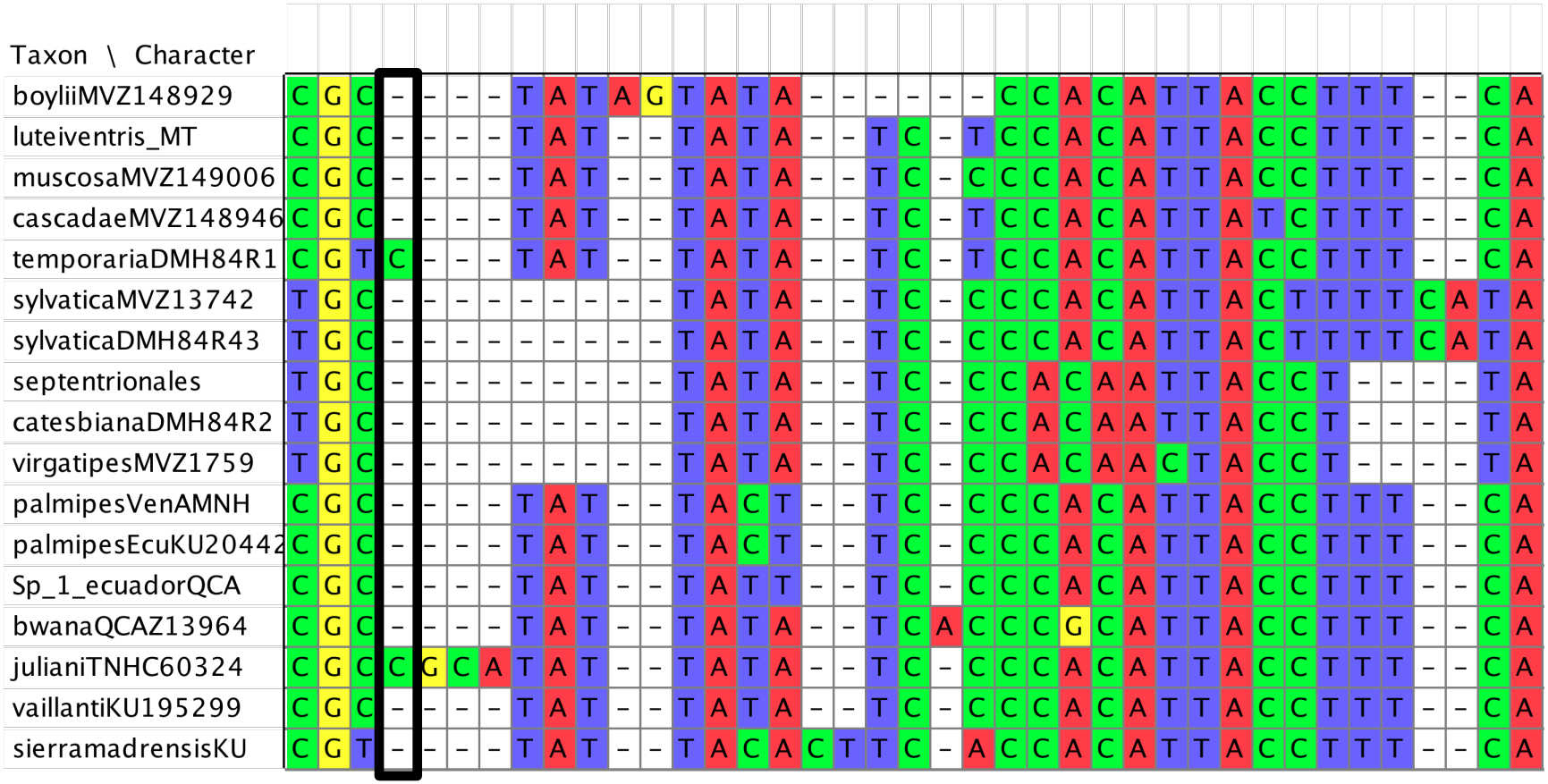
Gap model accuracy – 64 taxa, true alignments



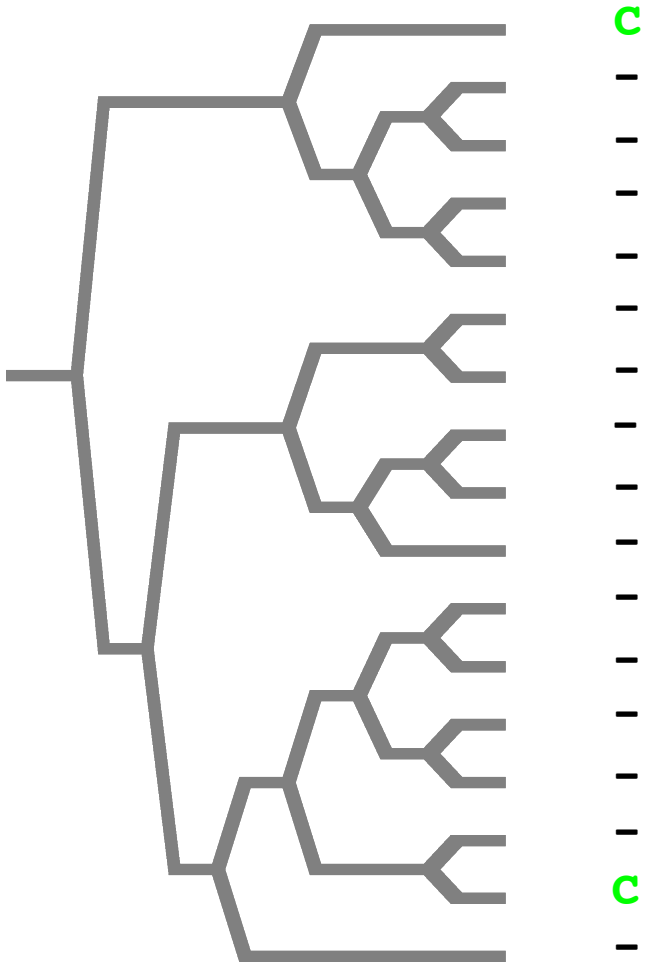
Gap model accuracy – 64 taxa, estimated alignments



Part of the problem: innocuous alignment errors

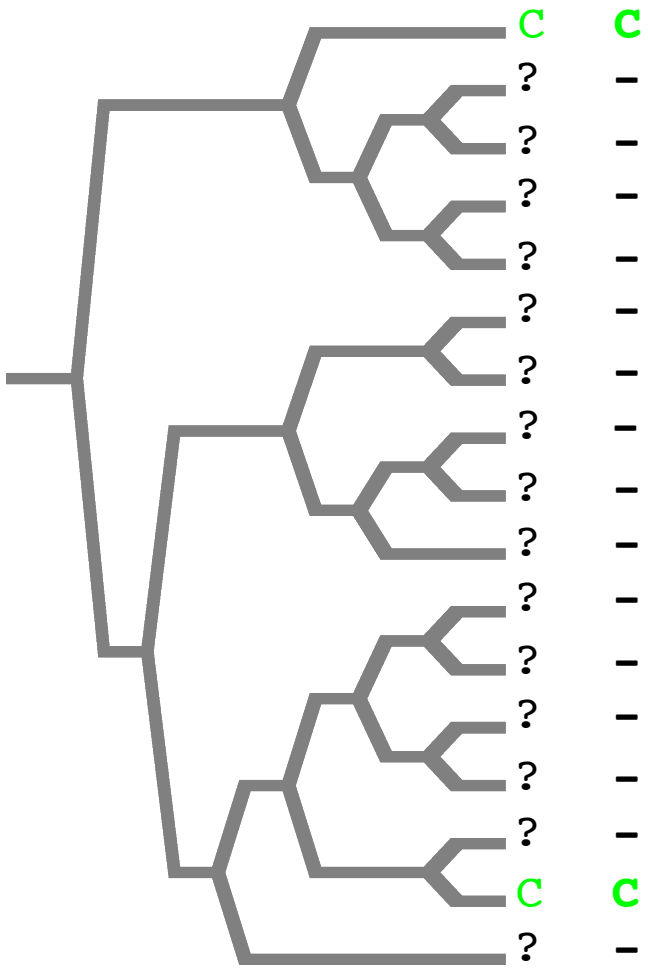


Innocuous alignment errors?



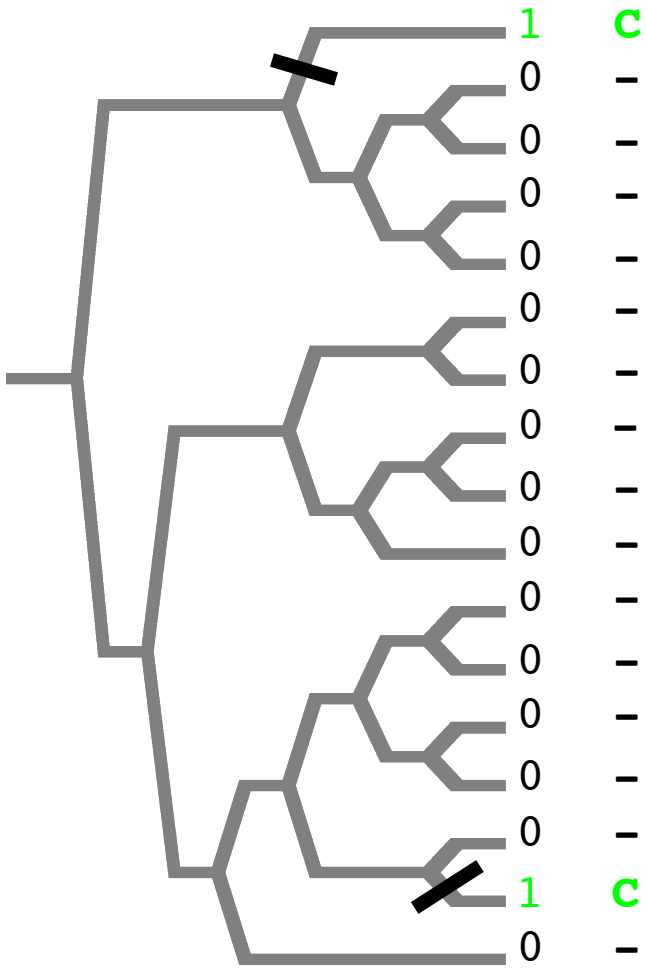
Innocuous alignment errors?

- GTR – no events, little effect



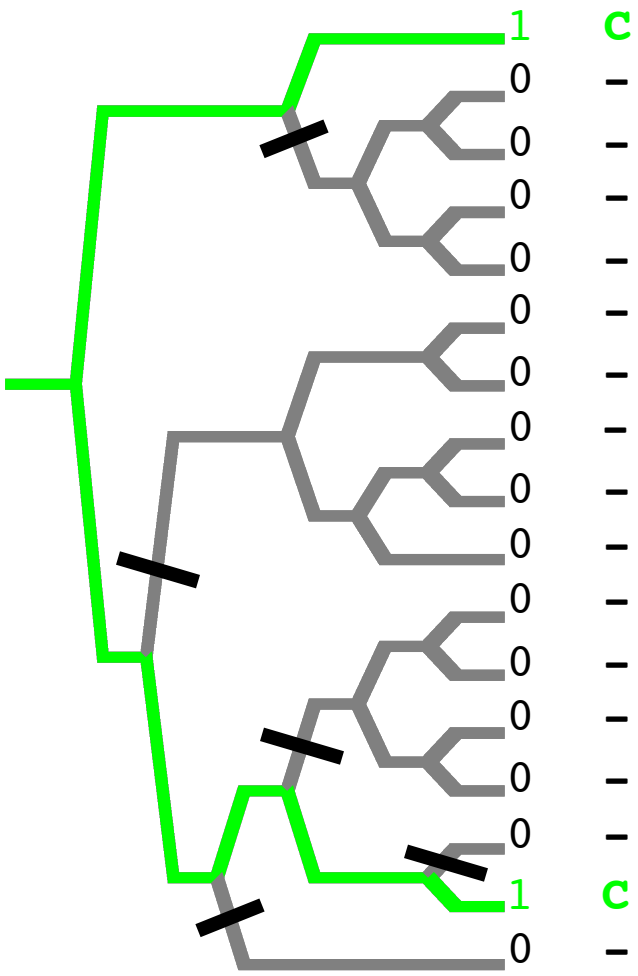
Innocuous alignment errors?

- GTR – no events, little effect
- Mk_v – 2 events, moderate effect



Innocuous alignment errors?

- GTR – no events, little effect
- Mk_v – 2 events, moderate effect
- DIMM – 5 deletions! strong effect (Dependent on tree size!!)



Conclusions: gap models

Indel events can provide useful signal *if* properly modeled

BUT, without correct column homology they can do more evil than good

Summary

- For >15 sequences, an unfathomably large number of possible trees are possible.
- We have to rely on heuristics that are not guaranteed to find the actual ("global") optimal solution.
- We have control on how thorough our searches are
- You should conduct many searches to look for evidence that your tree searching problem is difficult.
- GARLI, RAxML, ExaML, PhyML and PAUP* are the most commonly used ML tree searching for large datasets

Computer exercises