Tree Searching

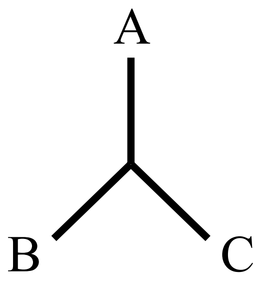
We've discussed how we rank trees

- Parsimony
- Least squares
- Minimum evolution
- Balanced minimum evolution
- Maximum likelihood (later in the course)

So we have ways of deciding what a good tree is when we see one, but . . .

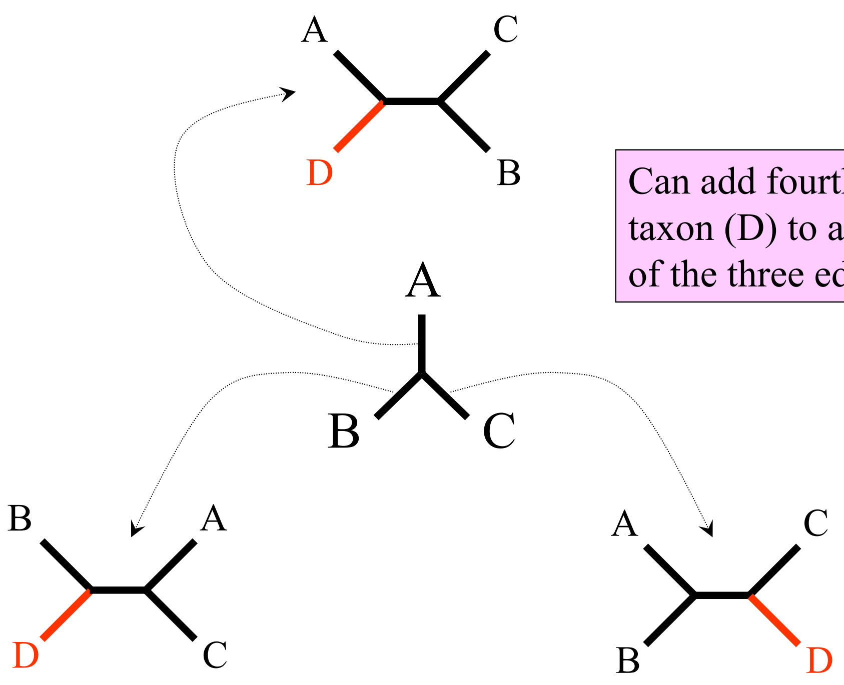
How do we find the best tree?
(or one that is good enough)

Exhaustive Enumeration



With the first three taxa, create the trivial unrooted tree

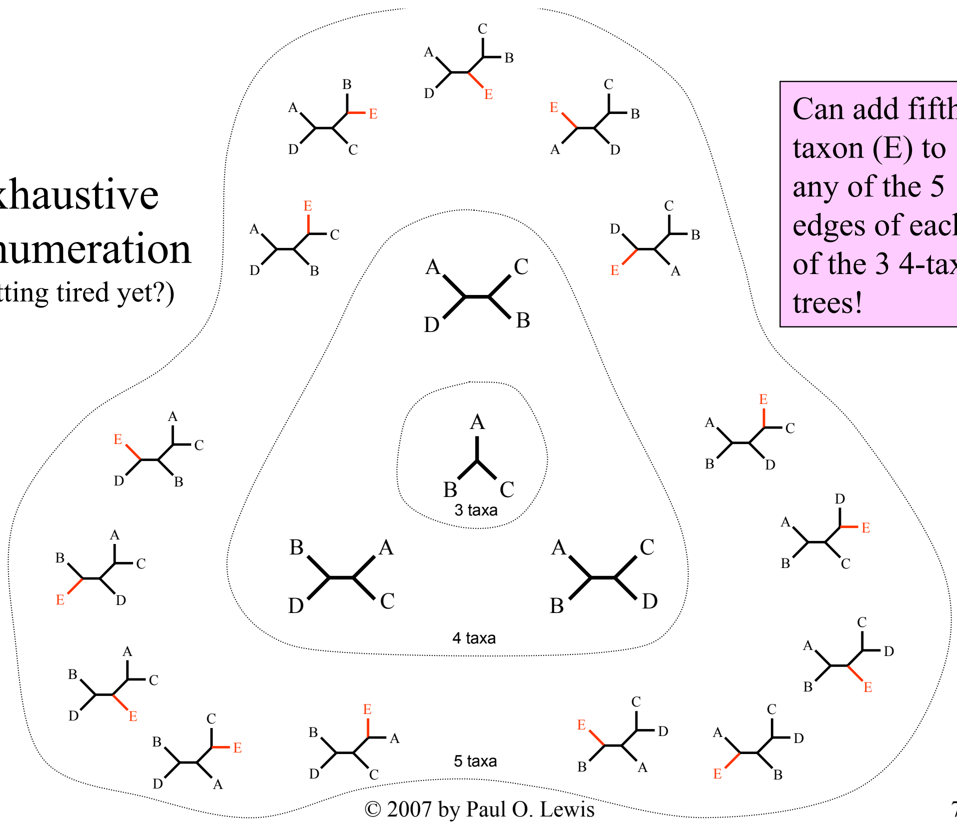
Exhaustive Enumeration...



Can add fourth taxon (D) to any of the three edges

Exhaustive Enumeration

(getting tired yet?)



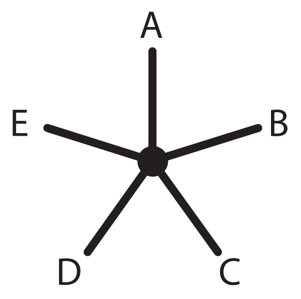
Can add fifth taxon (E) to any of the 5 edges of each of the 3 4-taxon trees!

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	

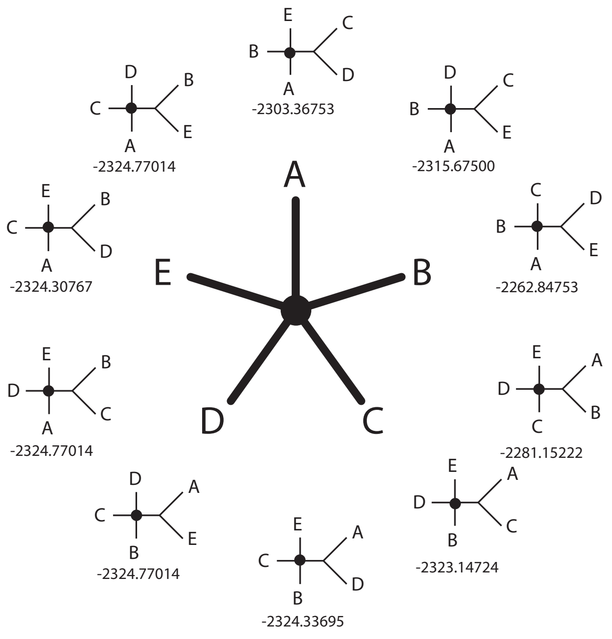
For N taxa:

$$\begin{aligned}
\#\text{ unrooted, binary trees} &= \prod_{i=3}^{N-1}(2i-3) \\
&= \prod_{i=4}^{N}(2i-5) \\
\#\text{ rooted, binary trees} &= \prod_{i=3}^{N}(2i-3) \\
&= (2N-3)(\#\text{ unrooted, binary trees})
\end{aligned}$$

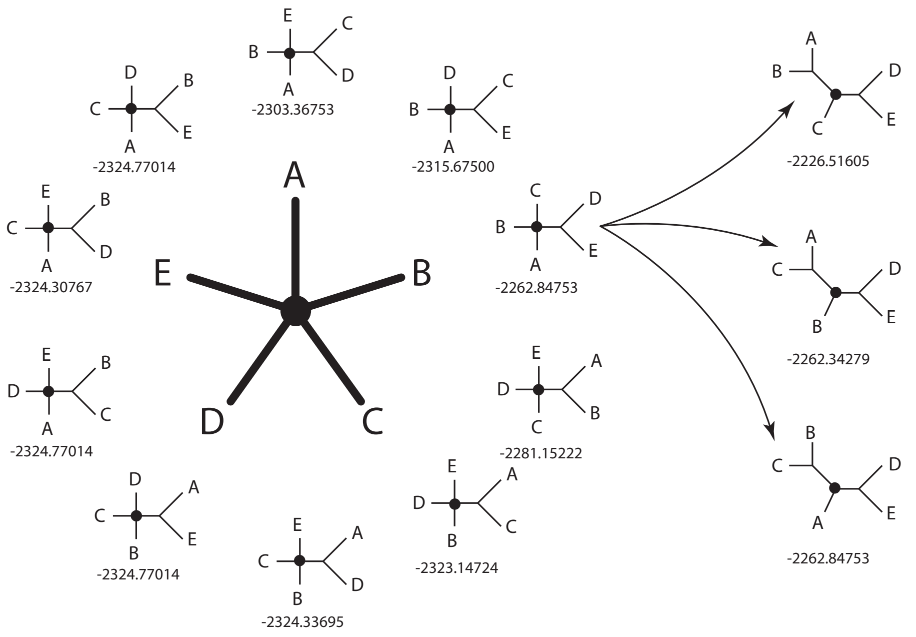
Star decomposition



Star decomposition



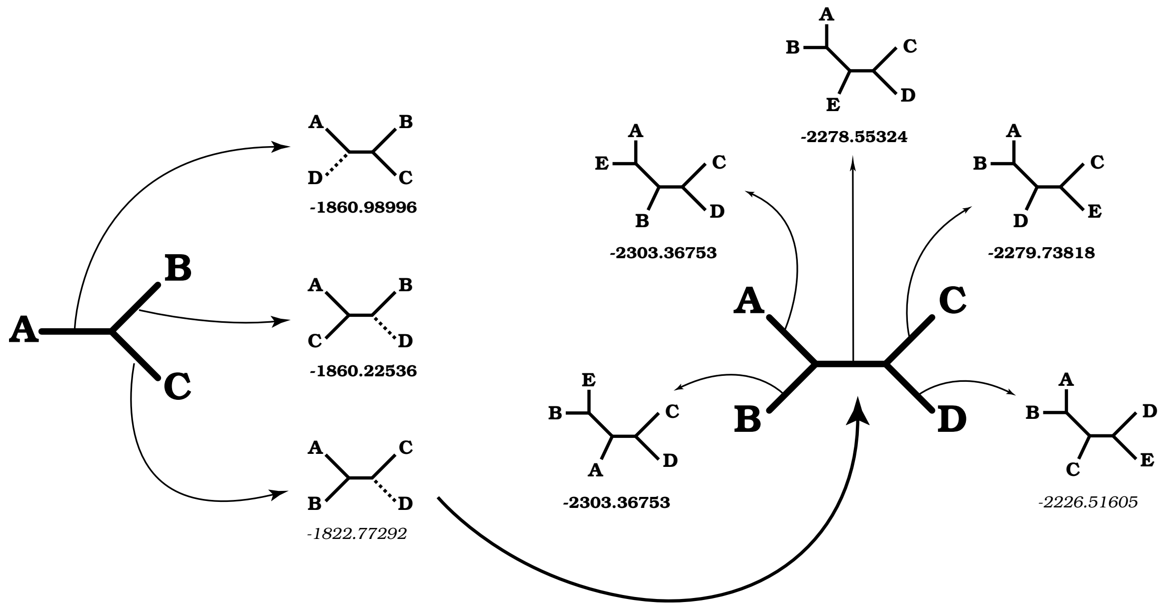
Star decomposition



Star decomposition

- Very "greedy" – it makes the best decision at each step, but does not try to "plan ahead". Once a pair of species are joined, they will not be separated.
- Neighbor-joining (Saitou and Nei, 1987) is star decomposition under the balanced minimum evolution criterion

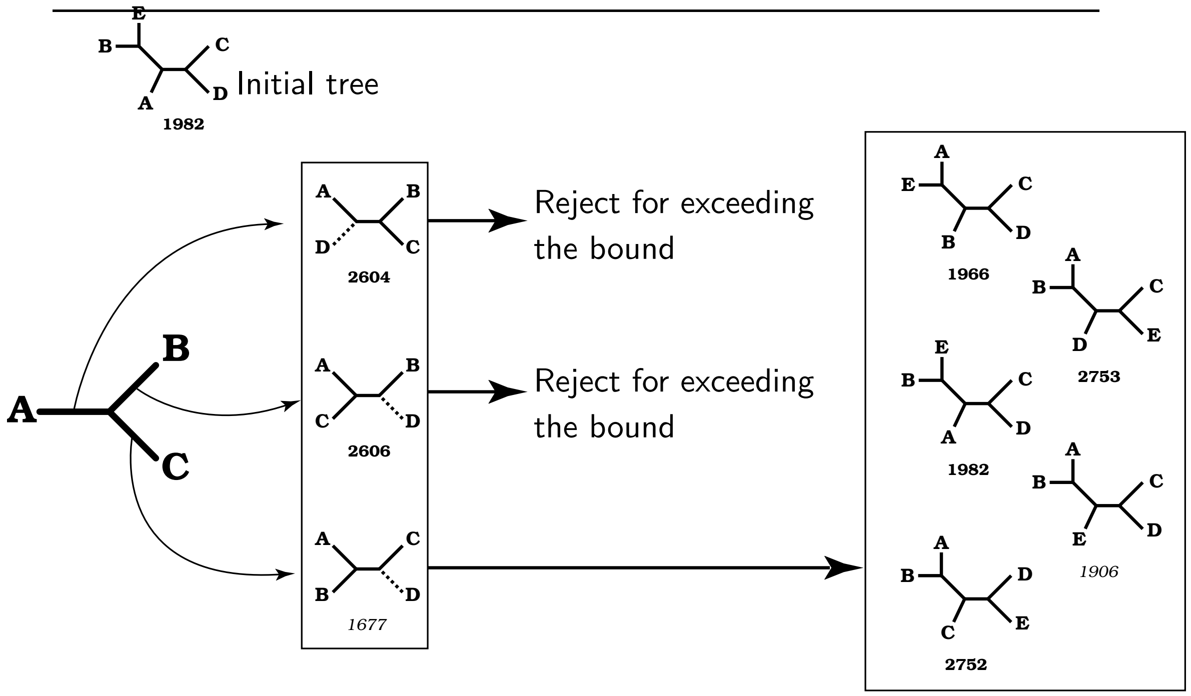
Stepwise addition



Stepwise addition

- Order-dependent (multiple random orderings can be used to give a range of starting trees for more thorough searches).
- Taxa joined initially may have intervening species added, but still fairly greedy.

Branch and bound



Initial tree
1982
2604
Reject for exceeding the bound
2606
Reject for exceeding the bound
1677
1966
2753
1982
1906
2752

Branch and bound

- Guaranteed to return the best tree(s)
- Typically only a viable option for < 30 species (depends on how clean the data is)

Trying to improve a tree

Neither stepwise addition nor star decomposition is guaranteed to return the best tree(s), but branch-and-bound (or exhaustive searching) is frequently infeasible.

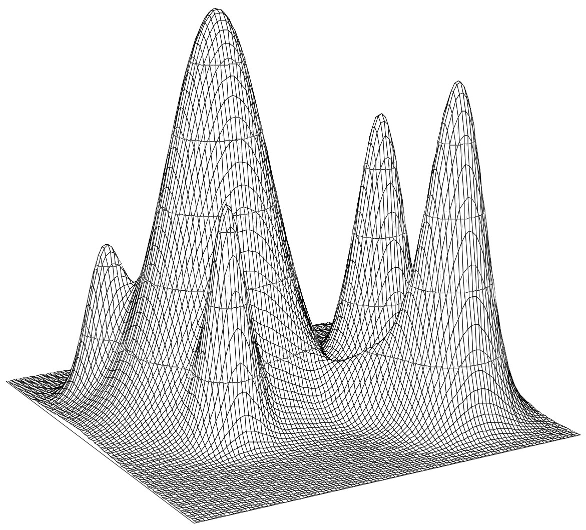
Heuristic hill-climbing searches can work quite well:

1. Start with a tree
2. Score the tree
3. Consider a new tree within the neighborhood of the current tree:
   (a) Score the new tree.
   (b) If the new tree has a better tree, use it as the "current tree"
   (c) Stop if there are no other trees within the neighborhood to consider.

These are **not** guaranteed to find even one of the optimal trees.

The most common way to explore the neighborhood of a tree is to swap the branches of the tree to construct similar trees.

Heuristics explore "Tree Space"

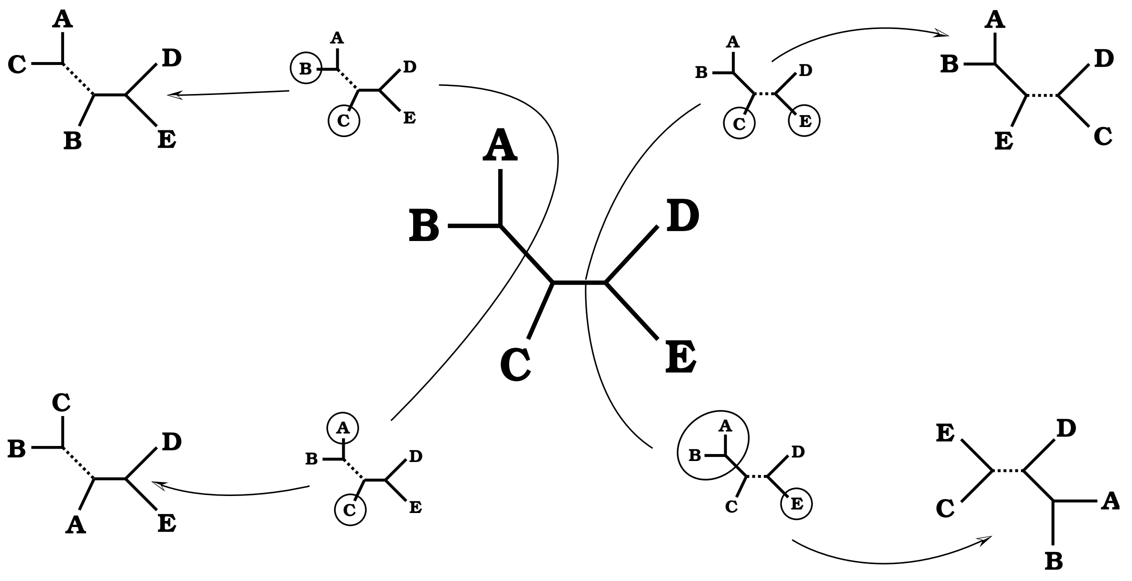


Most commonly used methods are "hill-climbers."

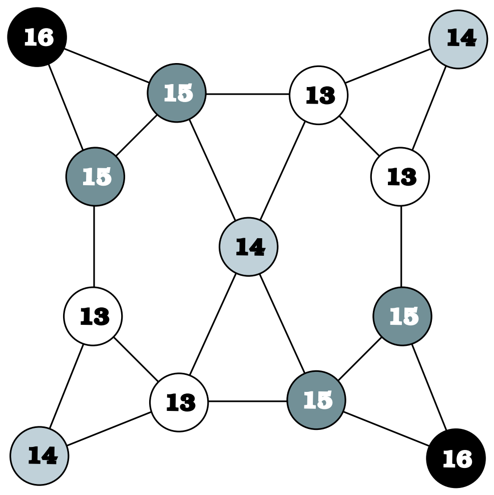
Multiple optima found by repeating searches from different origins.

Severity of the problem of multiple optima depends on step size.

Nearest Neighbor Interchange (NNI)



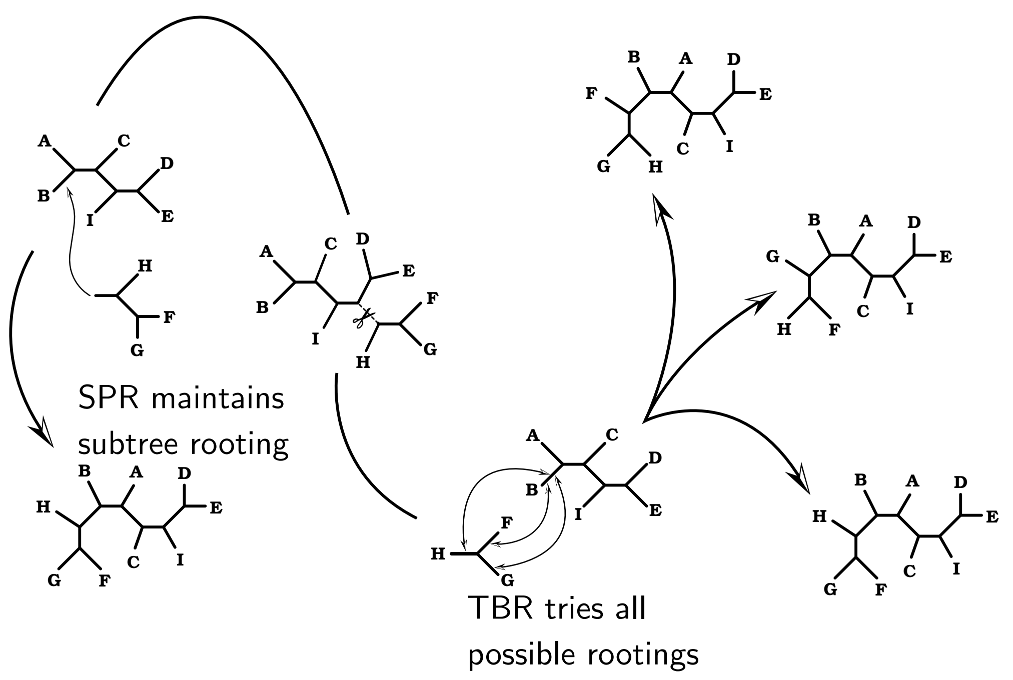
Nearest Neighbor Interchange (NNI)



1	A	T	C	G	C	A	G	G
2	A	T	T	G	G	T	G	A
3	G	G	C	T	C	A	C	G
4	A	T	C	T	G	T	C	G
5	G	G	T	T	C	T	G	A

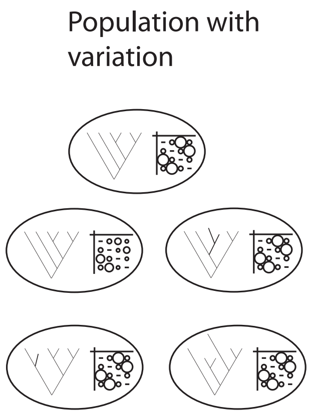
Contrived matrix with
2 NNI islands

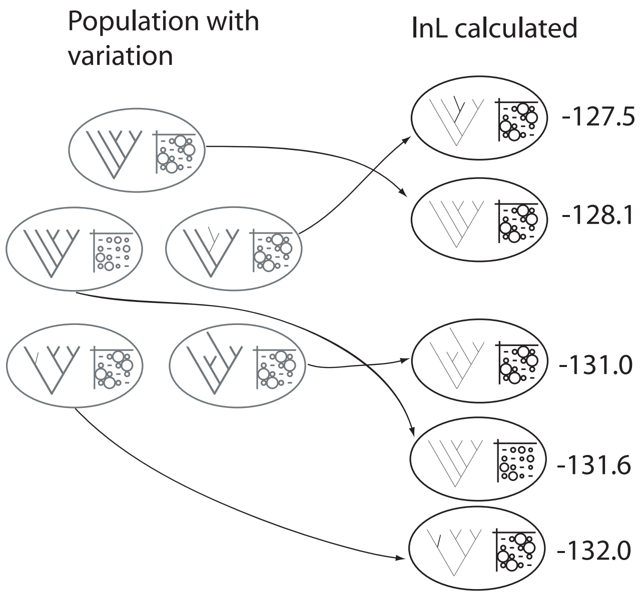
Subtree Pruning Regrafting (SPR) and Tree Bisection Reconnection (TBR)

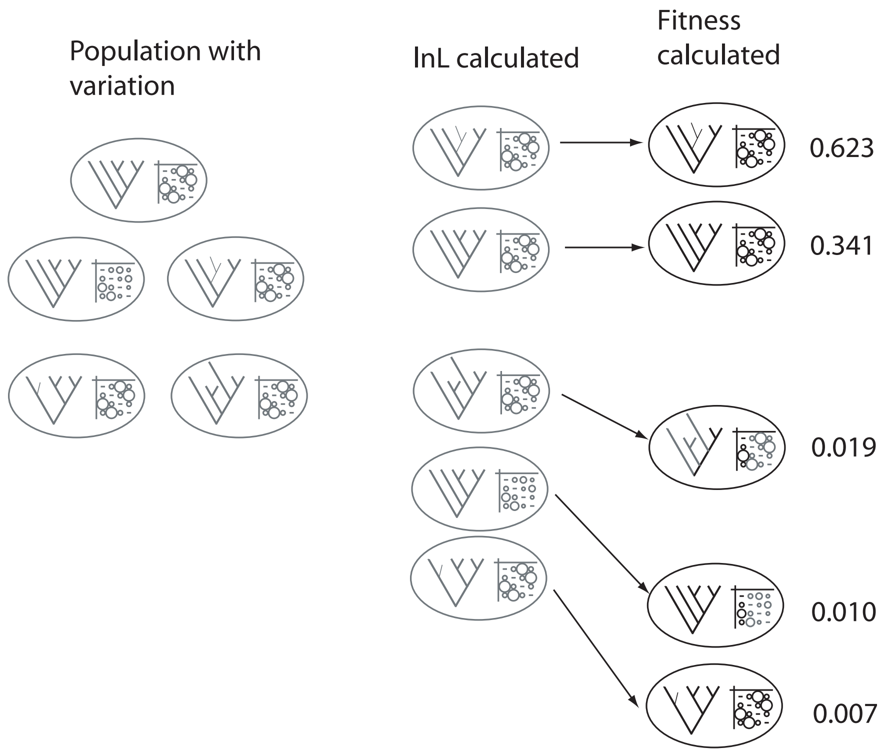

SPR maintains subtree rooting
TBR tries all possible rootings
A
B
C
D
E
F
G
H
I

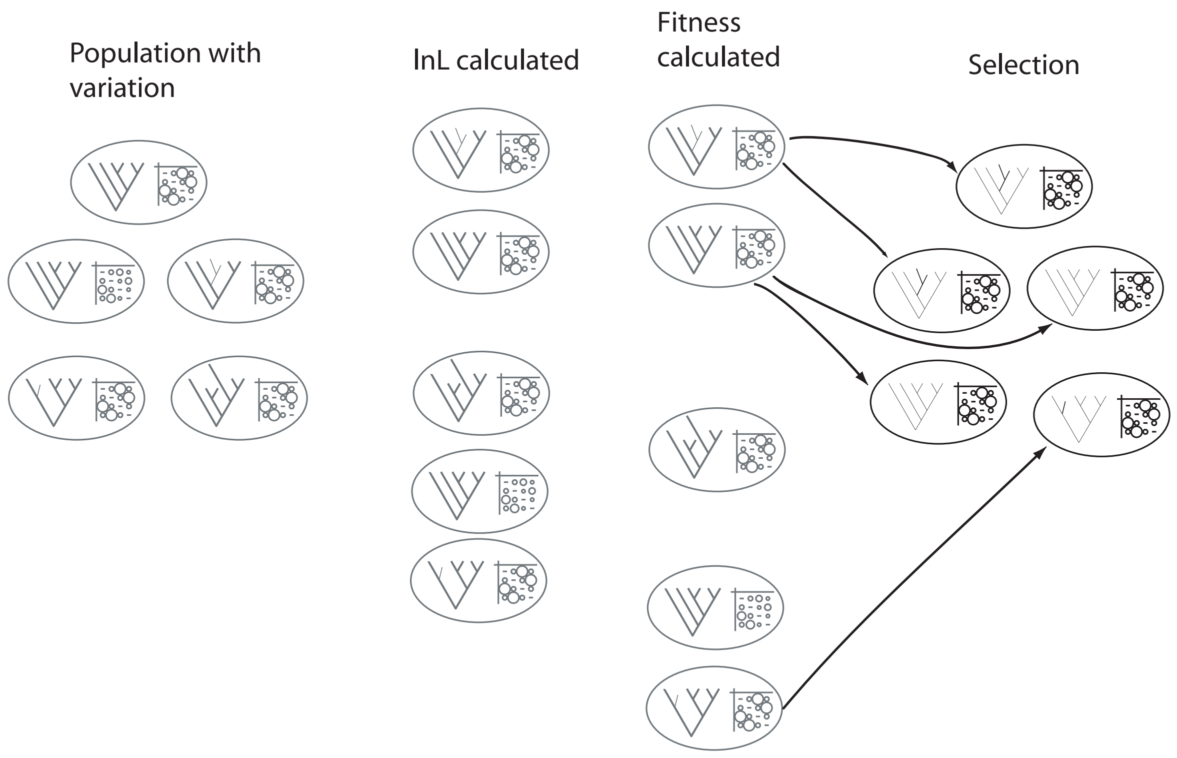
Many other heuristic strategies proposed

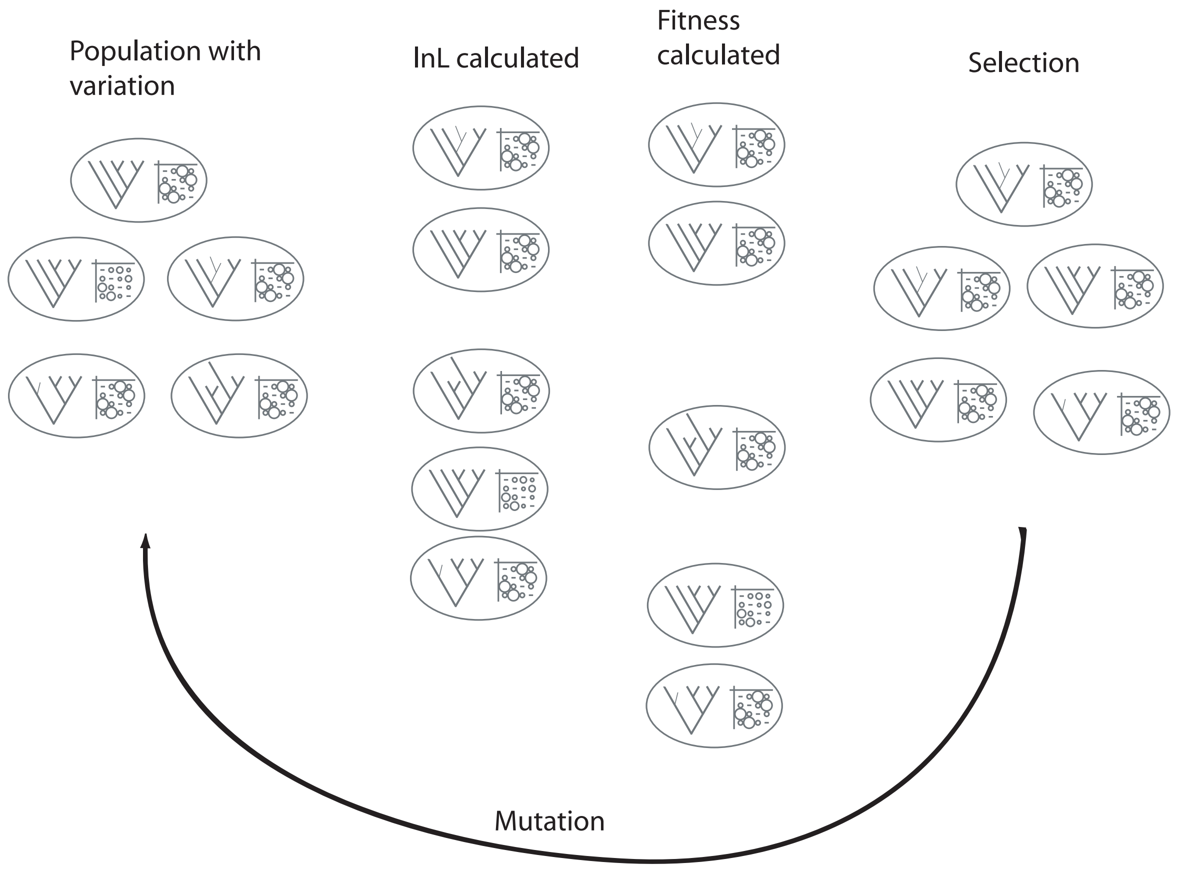
- Swapping need not include *all* neighbors (RAxML, `reconlimit` in PAUP*)
- "lazy" scoring of swaps (RAxML)
- Ignoring (at some stage) interactions between different branch swaps (PHYML)
- Stochastic searches
  - Genetic algorithms (GAML, MetaPIGA, GARLI)
  - Simulated annealing
- Divide and conquer methods (the sectortial searching of Goloboff, 1999; Rec-I-DCM3 Roshan 2004)
- Data perturbation methods (*e.g.* Kevin Nixon's "ratchet")


Population with variation


Population with variation
lnL calculated
-127.5
-128.1
-131.0
-131.6
-132.0


Population with variation
lnL calculated
Fitness calculated
0.623
0.341
0.019
0.010
0.007


Population with variation
lnL calculated
Fitness calculated
Selection



Software for searching under different criteria

Fast tree searching:

- Maximum likelihood – RAxML, FastTree. GARLI, phyml, Leaphy
- Distances – FastME (balanced minimum evolution); FastTree (profile approximation to balanced minimum evolution); PAUP (other distance-based criteria).
- Parsimony – TNT

Conclusions on searching

1. The large number of trees make it infeasible to evaluate every tree;
2. Intuitive, hill climbing routines often perform well;
3. Repeated searching from multiple starting points helps give you a sense of how difficult searching is for your dataset.
4. The ease of tree searching is a separate issue from statistical support. Well-supported clades are often easy to find, but we do **not** simply use the repeatability of a trees in independent searches as a measure of support (we'll talk about assessing support tomorrow).

References

Saitou, N. and Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425.