Phylogenetic trees III
Maximum Parsimony

Gerhard Jäger

ESSLLI 2016

Background

Character-based tree estimation

- distance-based tree estimation has several drawbacks:
  - very strong theoretical assumptions - e.g., all characters evolve at the same rate
  - Neighbor Joining and UPGMA produce good but sub-optimal trees
  - no solid statistical justification for those algorithms
  - distances are black boxes — we get a tree, but we learn nothing about the history of individual characters
- **character-based** tree estimation
  - estimates complete scenario (or distribution over scenarios) for each character
  - finds the tree that best explains the observed variation in the data (at least in theory, that is...)

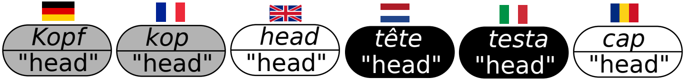
Parsimony

Parsimony of a tree

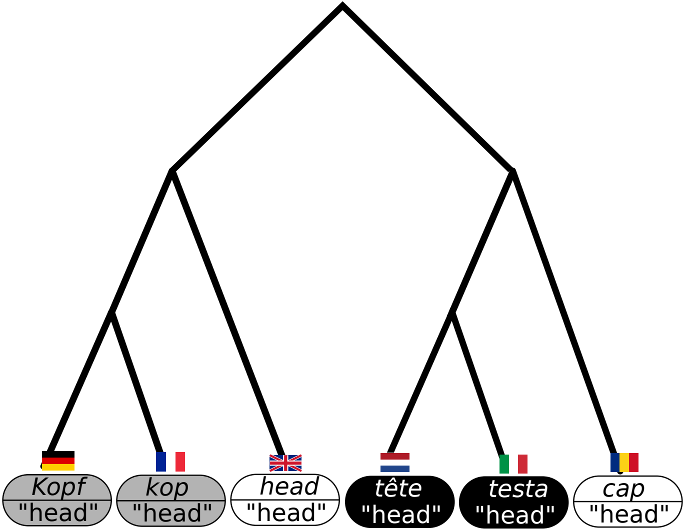
background reading: Ewens and Grant (2005), 15.6

- suppose a character matrix and a tree are given
- **parsimony score:** minimal number of mutations that has to be assumed to explain the character values at the tips, given the tree

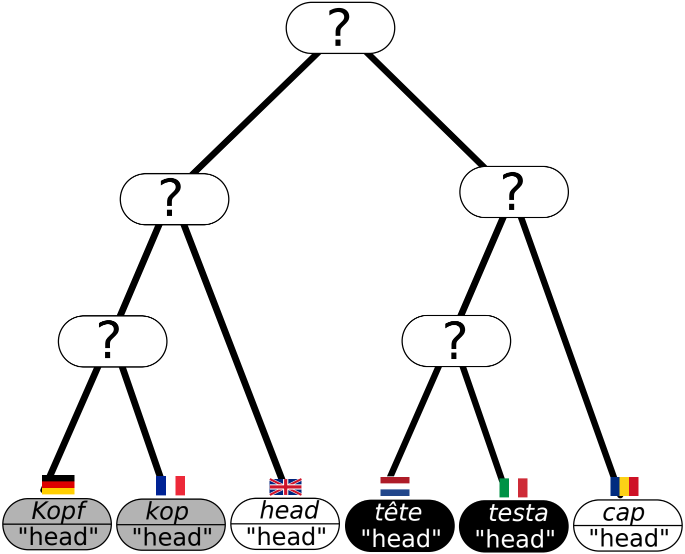
Parsimony of a tree



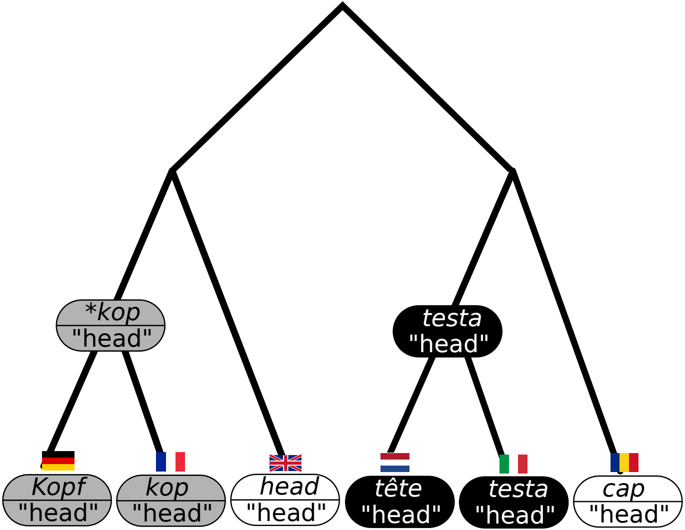
Parsimony of a tree



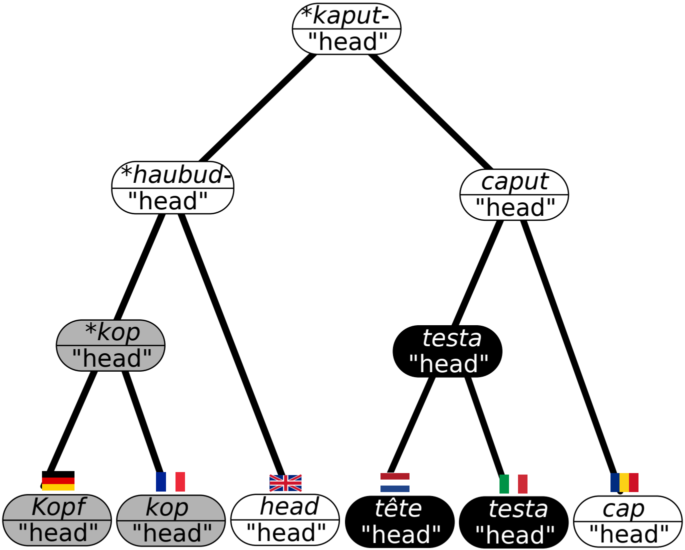
Parsimony of a tree



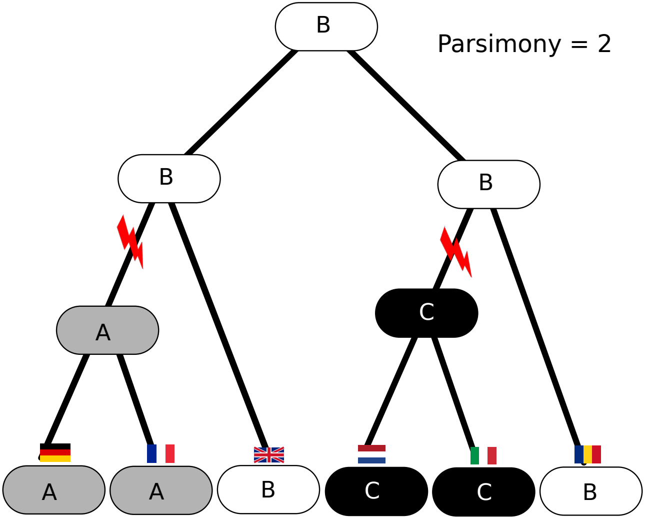
Parsimony of a tree



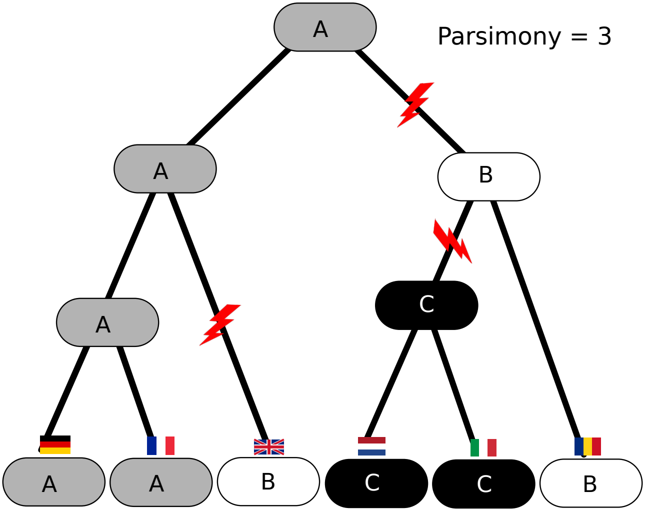
Parsimony of a tree



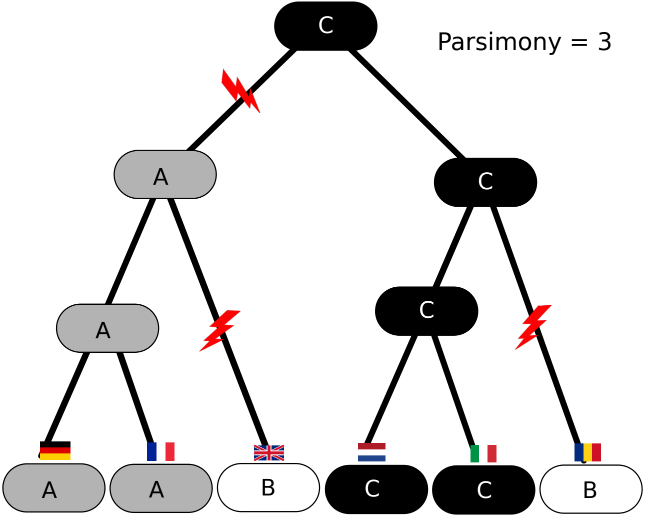
Parsimony reconstruction



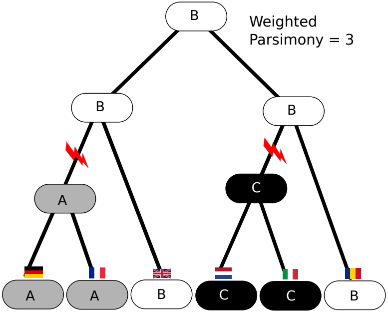
Parsimony reconstruction



Parsimony reconstruction



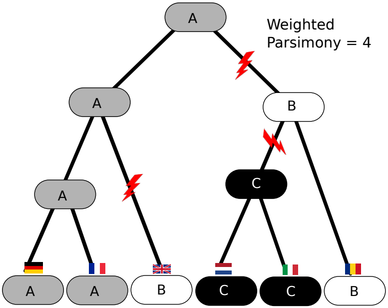
Weighted parsimony reconstruction



Weight matrix

	A	B	C
A	0	1	2
B	1	0	2
C	2	2	0

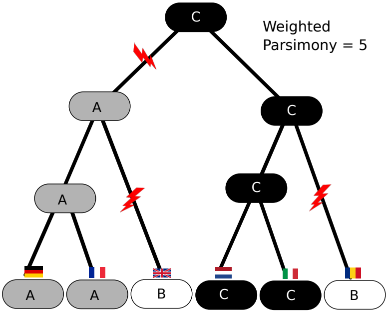
Weighted parsimony reconstruction



Weight matrix

	A	B	C
A	0	1	2
B	1	0	2
C	2	2	0

Weighted parsimony reconstruction

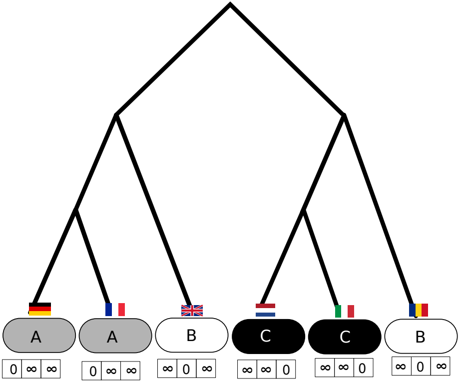


Weight matrix

	A	B	C
A	0	1	2
B	1	0	2
C	2	2	0

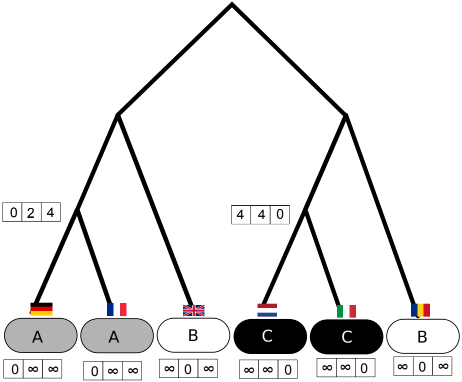
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



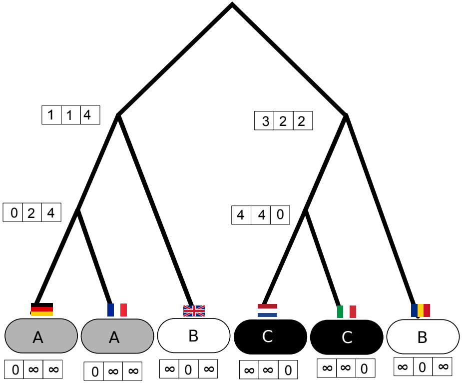
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



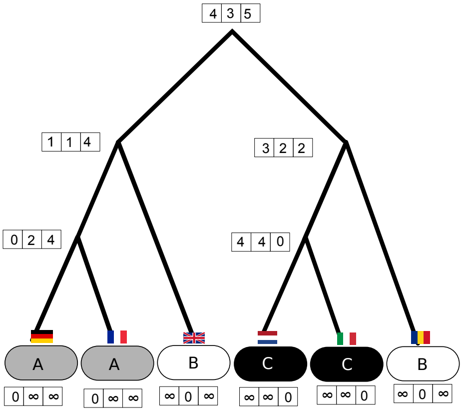
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



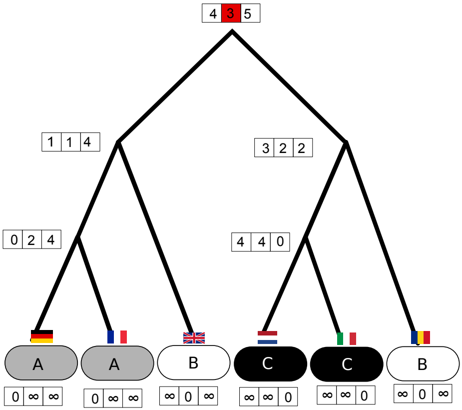
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



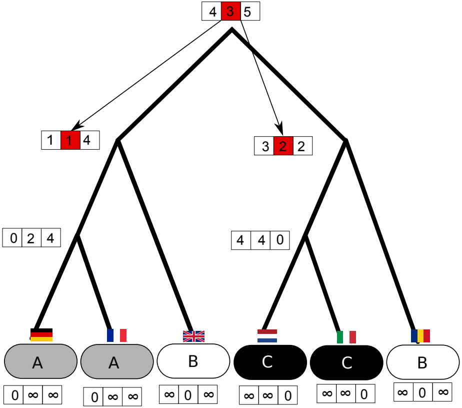
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



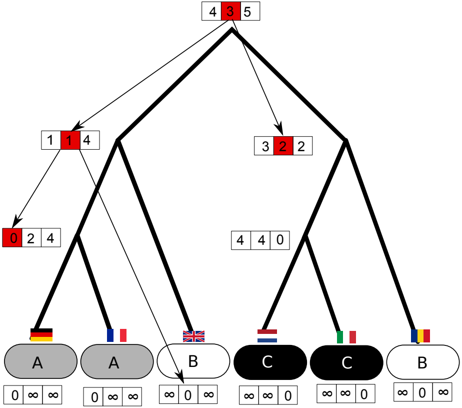
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



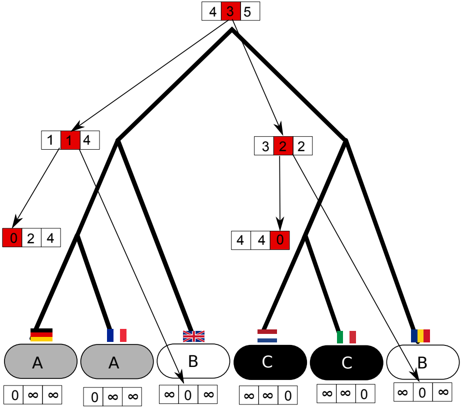
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



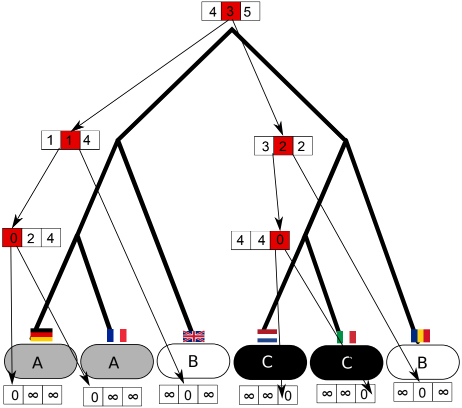
Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



Dynamic Programming (Sankoff Algorithm)

$$\text{wp}(\text{mother}, s) = \sum_{d \in \text{daughters}} \min_{s' \in \text{states}} (w(s, s') + \text{wp}(d, s'))$$



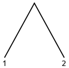
Searching for the best tree

- total parsimony score of tree: sum over all characters
- note: if weight matrix is symmetric, location of the root doesn't matter
- Sankoff algorithm efficiently computes parsimony score of a given tree
- goal: tree which minimizes parsimony score
- no efficient way to find the optimal tree $\rightarrow$ heuristic tree search

Searching the tree space

How many rooted tree topologies are there?

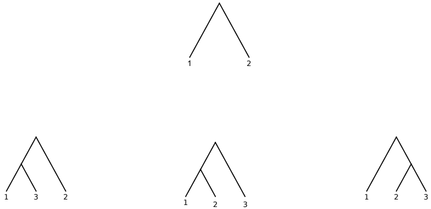
n=2



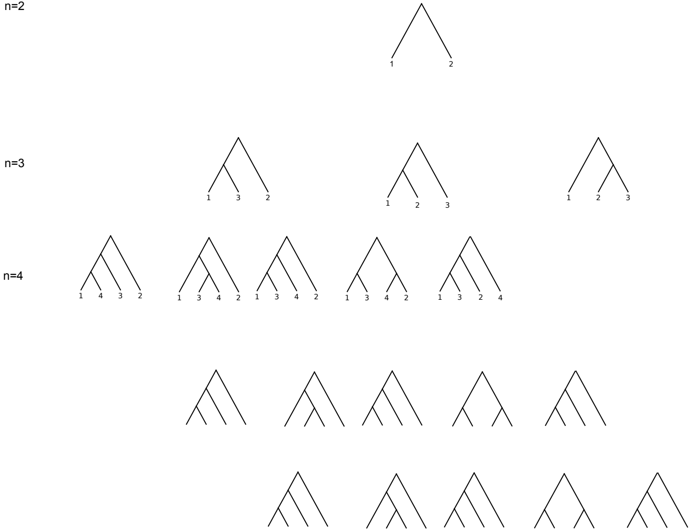
How many rooted tree topologies are there?

n=2

n=3



How many rooted tree topologies are there?



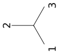
How many rooted tree topologies are there?

$$f(2) = 1$$
$$f(n+1) = (2n-3)f(n)$$
$$f(n) = \frac{(2n-3)!}{2^{n-2}(n-2)!}$$

2	1
3	3
4	15
5	105
6	945
7	10395
8	135135
9	2027025
10	34459425
11	654729075
12	13749310575
13	316234143225
14	$7.9e+12$
15	$2.1e+14$
16	$6.1e+15$
17	$1.9e+17$
18	$6.3e+18$
19	$2.2e+20$
20	$8.2e+21$
21	$3.1e+23$
22	$1.3e+25$
23	$5.6e+26$
24	$2.5e+28$
25	$1.1e+30$
26	$5.8e+31$
27	$2.9e+33$
28	$1.5e+35$
29	$8.6e+36$
30	$4.9e+38$
31	$2.9e+40$
32	$1.7e+42$
33	$1.1e+44$
34	$7.2e+45$
35	$4.8e+47$
36	$3.3e+49$
37	$2.3e+51$
38	$1.7e+53$
39	$1.3e+55$
40	$1.0e+57$

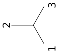
How many unrooted tree topologies are there?

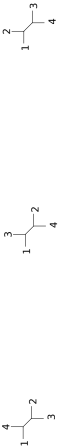
n=3



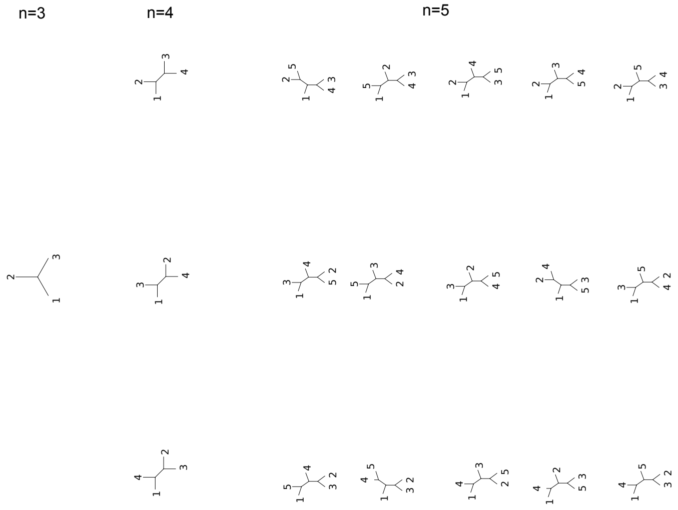
How many unrooted tree topologies are there?

n=3 n=4





How many unrooted tree topologies are there?



How many unrooted tree topologies are there?

$$f(3) = 1$$
$$f(n+1) = (2n-3)f(n)$$
$$f(n) = \frac{(2n-5)!}{2^{n-3}(n-3)!}$$

3	1
4	3
5	15
6	105
7	945
8	10395
9	135135
10	2027025
11	34459425
12	654729075
13	13749310575
14	316234143225
15	$7.90e+12$
16	$2.13e+14$
17	$6.19e+15$
18	$1.91e+17$
19	$6.33e+18$
20	$2.21e+20$
21	$8.20e+21$
22	$3.19e+23$
23	$1.31e+25$
24	$5.63e+26$
25	$2.53e+28$
26	$1.19e+30$
27	$5.84e+31$
28	$2.98e+33$
29	$1.57e+35$
30	$8.68e+36$
31	$4.95e+38$
32	$2.92e+40$
33	$1.78e+42$
34	$1.12e+44$
35	$7.29e+45$
36	$4.88e+47$
37	$3.37e+49$
38	$2.39e+51$
39	$1.74e+53$
40	$1.31e+55$

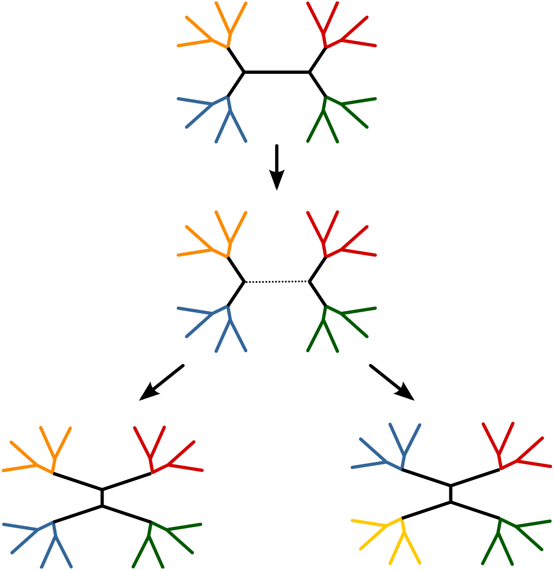
Heuristic tree search

- tree space is too large to do an exhaustive search if n (number of taxa) is larger than 12 or so
- heuristic search:
  - start with some tree topology (e.g., Neighbor-Joining tree)
  - apply a bunch of local modifications to the current tree
  - if one of the modified tree has lower or equal parsimony, move to that tree
  - stop if no further improvement is possible
- $\Rightarrow$ standard approach for optimization problems in computer science

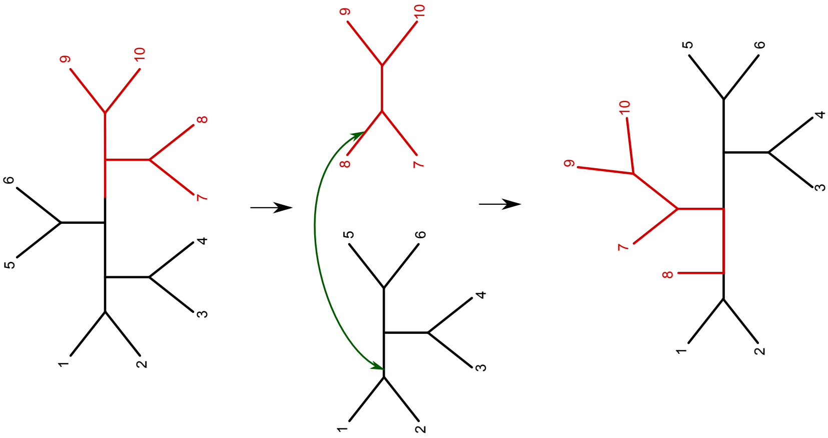
Tree modifications

- three tree modifications commonly in use:
  1. *Nearest Neighbor Interchange* (NNI)
  2. *Tree Bisection and Reconnection* (TBR)
  3. *Subtree Pruning and Regrafting* (SPR)
- local modifications are better than arbitrary moves in tree space because partial parsimony computations can be re-used in modified tree

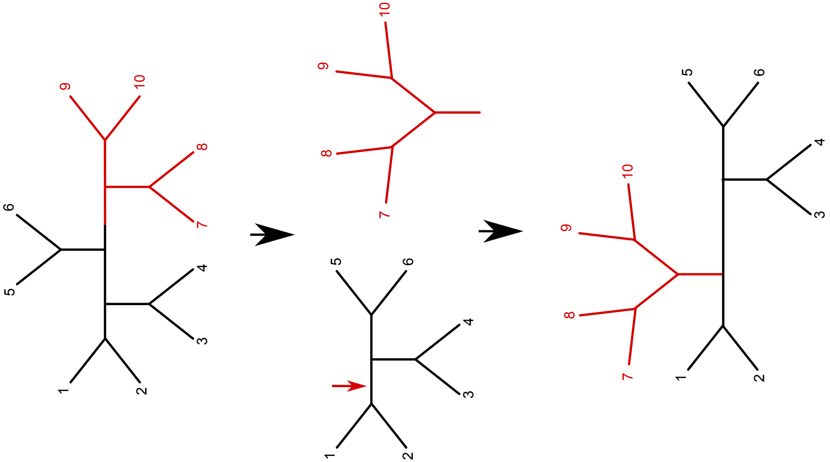
Nearest Neighbor Interchange



Tree Bisection and Reconection



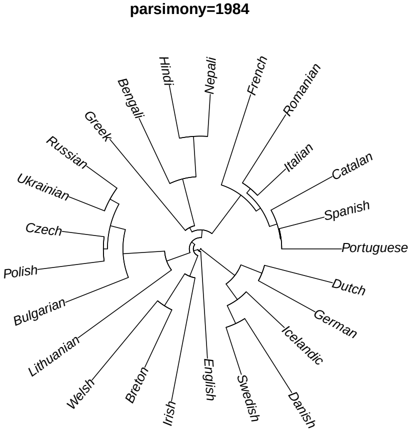
Subtree Pruning and Regrafting



Heuristic tree search

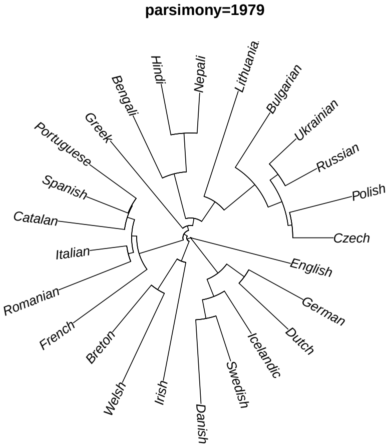
- NNI is very local $\rightarrow$ only $\mathcal{O}(n)$ possible moves
- SPR and TBR are more aggressive $\rightarrow$ $\mathcal{O}(n^2)/\mathcal{O}(n^3)$ possible moves
- NNI search is comparatively fast, but prone to get stuck in local optima

Running example: SPR search with cognate data

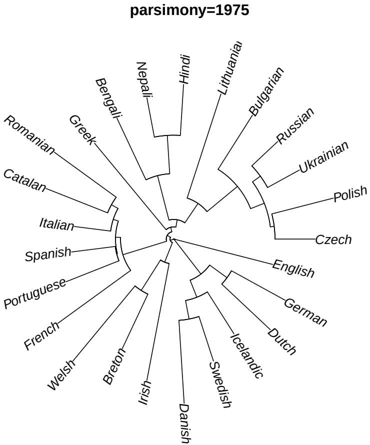


starting with Neighbor Joining tree ...

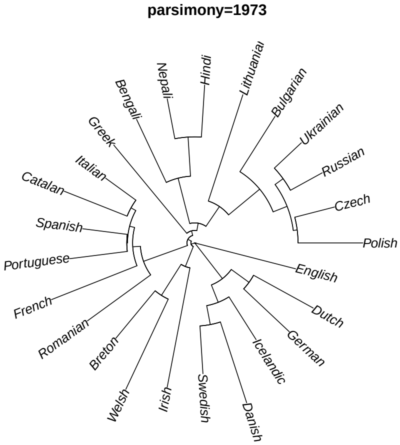
Running example: SPR search with cognate data



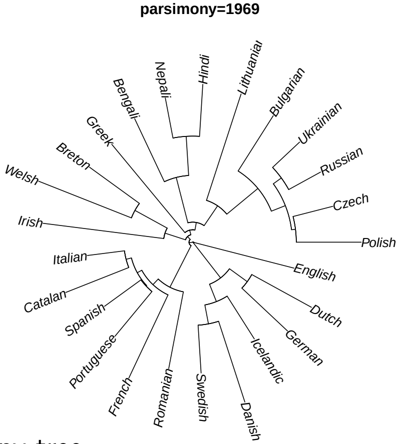
Running example: SPR search with cognate data



Running example: SPR search with cognate data



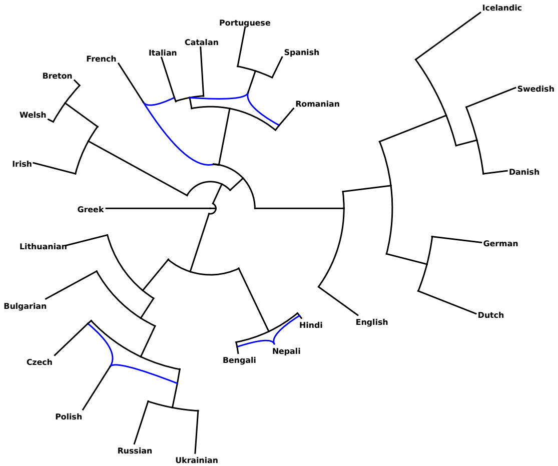
Running example: SPR search with cognate data



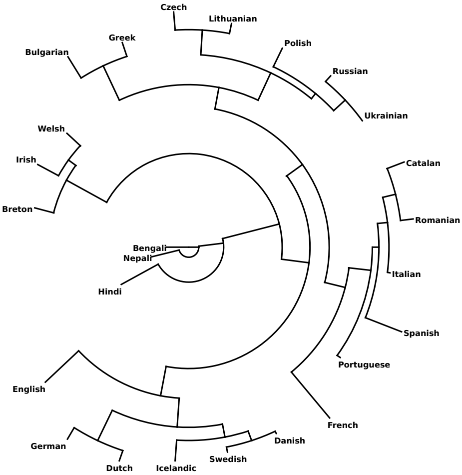
. . . Maximum Parsimony tree

Running example: SPR search with cognate data

- there are actually 16 different trees with minimal parsimony score



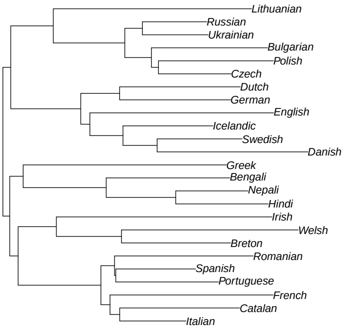
MP tree for WALS characters



MP tree for sound-concept characters

Dollo parsimony

- previous trees were estimated with a symmetric weight matrix
- if weights are asymmetric, location of the root matters
- extreme case: **Dollo Parsimony**
- $w(0 \rightarrow 1) = \infty$



Maximum Parsimony: Discussion

- Once we have found the best tree (or, in any event, which is very close to the best tree), we can reconstruct ancestral states via the Sankoff algorithm
- this allows to compute statistics about stability of characters, frequency and location of parallel changes etc.
  - $\Rightarrow$ much more informative than distance-based inference

Maximum Parsimony: Discussion

- disadvantages of MP:
  - simulation studies: capacity to recover the true tree is decent but not overwhelming
  - possibility of multiple mutations on a single branch is not taken into consideration
  - all characters are treated equal; no discrimination between stable and volatile characters
  - ties are common, especially if you have few data
  - values for weight matrix are *ad hoc*
  - no real theoretical justification
    - Why should the true tree minimize the total number of mutations?
    - Rests on a valid intuition: Mutations are unlikely, so assuming fewer mutations increases the likelihood of the data.
    - Likelihood is not formally derived from a probabilistic modell though.

Next step: Maximum *Likelihood* tree estimation

Hands on

- Install the software Paup*.
- Go to the directory where you have the put the nexus files and type
  `> paup4 ielex.bin.nex`
- At Paup's command prompt, type
  `paup> hsearch.`
- Display tree with
  `paup> describetree /plot=phylo`
- Save result with
  ```
  paup> savetree format=newick file = ielex.mp.tre \
  brlen=yes
  ```
- Leave Paup* with
  `paup> q`
- Install `Dendroscope` or `FigTree` and load `ielex.mp.tre`.

Ewens, W. and G. Grant (2005). *Statistical Methods in Bioinformatics: An Introduction*. Springer, New York.