CS 68: BIOINFORMATICS

Prof. Sara Mathieson
Swarthmore College
Spring 2018

Outline: Mar 2

- Introduce Lab 5
- Recap theory of Neighbor-Joining
- Begin: parsimony (Fitch's algorithm)

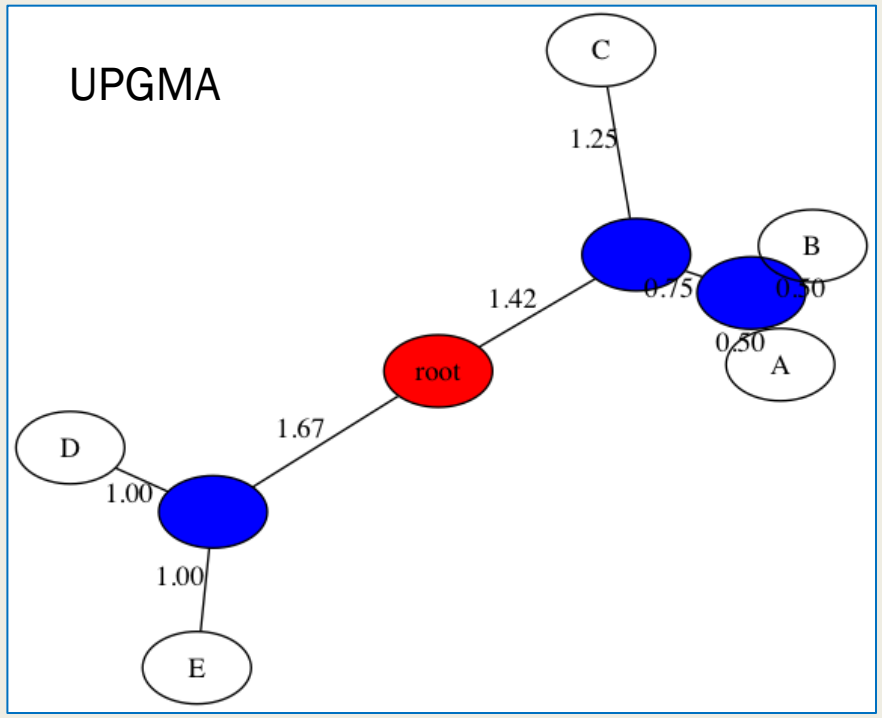
Notes:

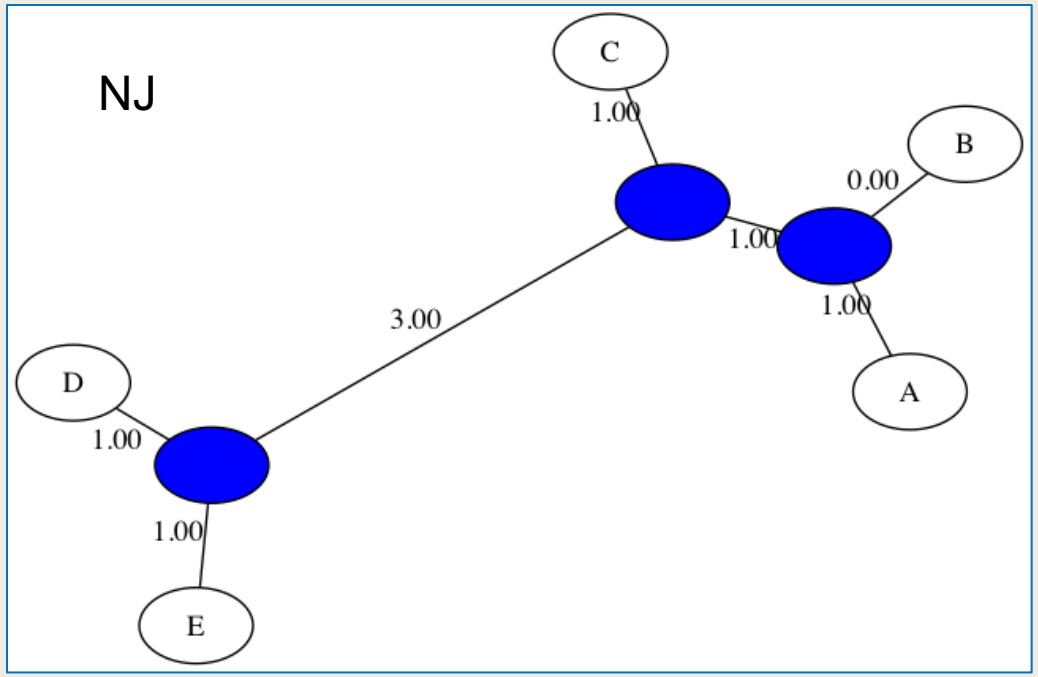
- Meet with your partner to begin Lab 5
- I am around this afternoon

Lab 5 introduction

Lab 5

- Goals: implement both UPGMA and NJ
- Analyze the trees produced by each one





Example output images

Lab 5

- Using pygraphviz

```
import pygraphviz as gv
tree = gv.AGraph() # constructs a graph object
tree.add_node("A") # the string is both the label and hash key
tree.add_node("B")
tree.add_edge("A", "B", label="1.0", len=1.0) # set string label as length
tree.draw("my_tree.png",prog="neato") # neato does node/edge layout
```

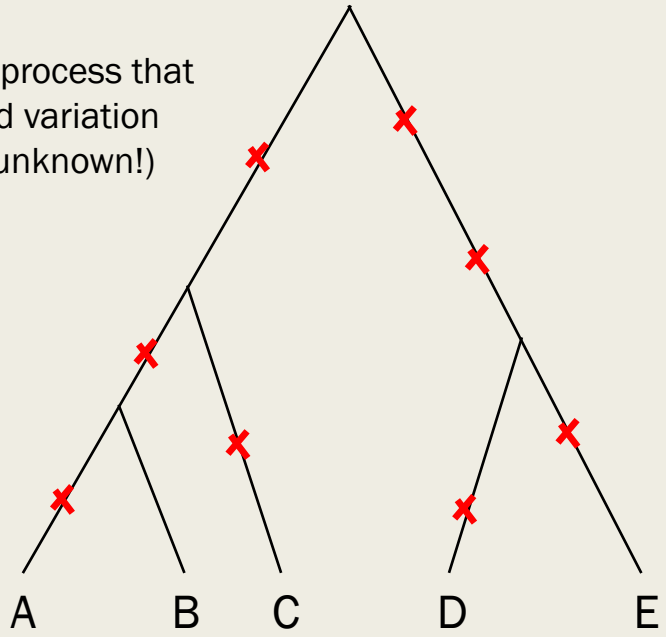
- Checkpoint next Thursday (completed UPGMA)

- Extra credit opportunities (rare!)
  - *Figure out how to layout UPGMA trees so the root is at the top and leaves at the bottom*
  - *Analyze the induced metrics produced by UPGMA and NJ*

Recap Neighbor-Joining Theory

What have we learned from our example?

Mutational process that generated variation (usually unknown!)

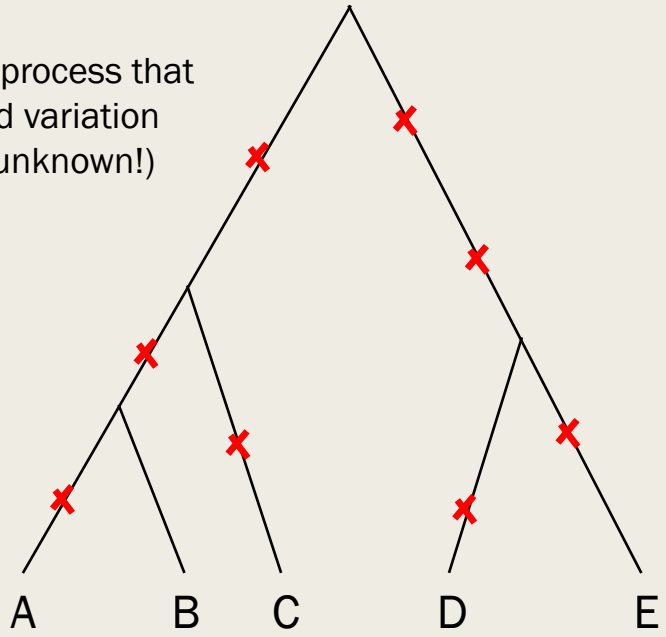


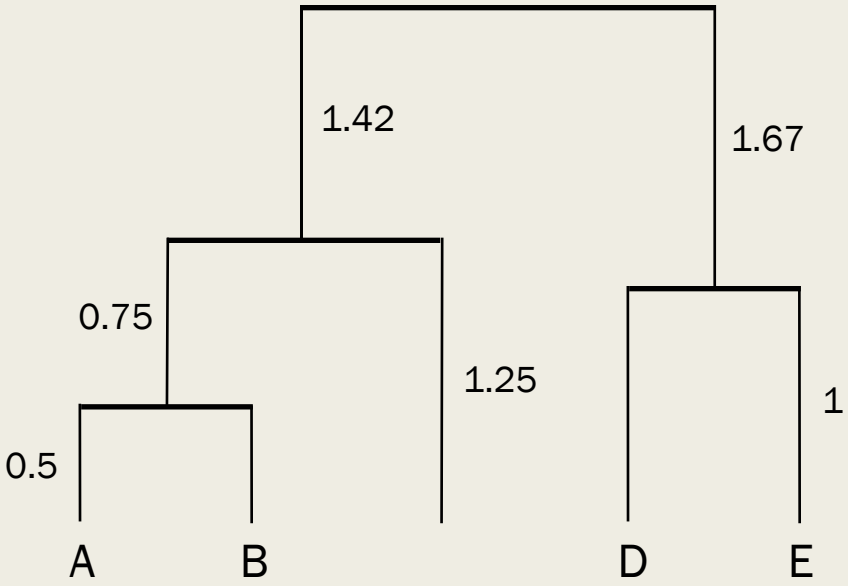
Original input dissimilarity map

δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

What have we learned from our example?

Mutational process that generated variation (usually unknown!)





Tree produced by UPGMA (rooted)

Original input dissimilarity map

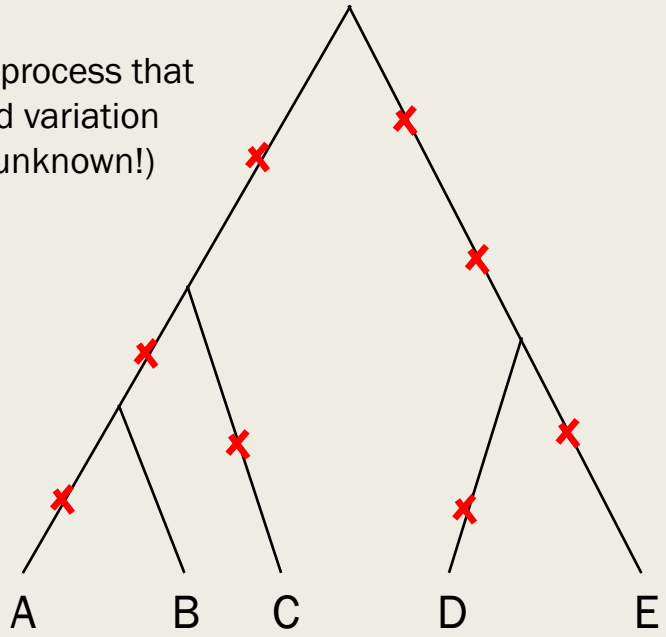
δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

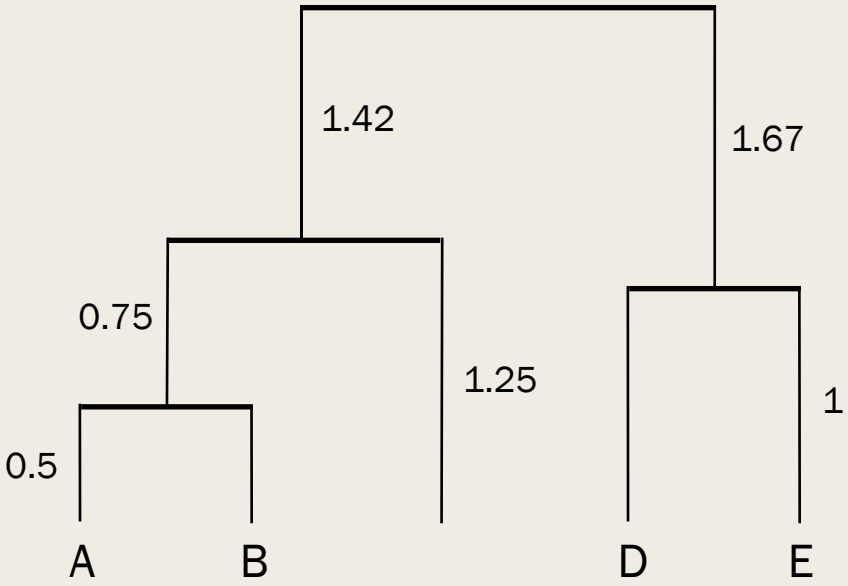
δ_{UPGMA}	A	B	C	D	E
A	0	1	2.5	5.33	5.33
B		0	2.5	5.33	5.33
C			0	5.33	5.33
D				0	2
E					0

Tree metric on *X* induced by UPGMA

What have we learned from our example?

Mutational process that generated variation (usually unknown!)





Tree produced by UPGMA (rooted)

Original input dissimilarity map

δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

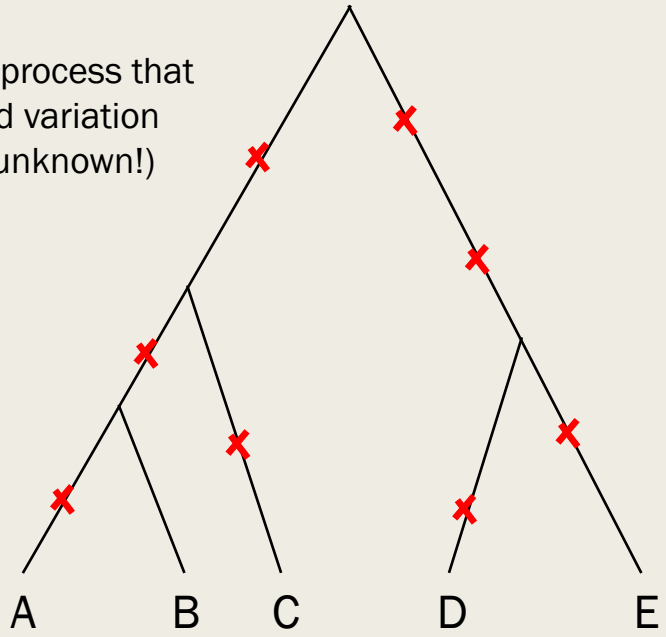
$\neq$

δ_{UPGMA}	A	B	C	D	E
A	0	1	2.5	5.33	5.33
B		0	2.5	5.33	5.33
C			0	5.33	5.33
D				0	2
E					0

Tree metric on *X* induced by UPGMA

What have we learned from our example?

Mutational process that generated variation (usually unknown!)

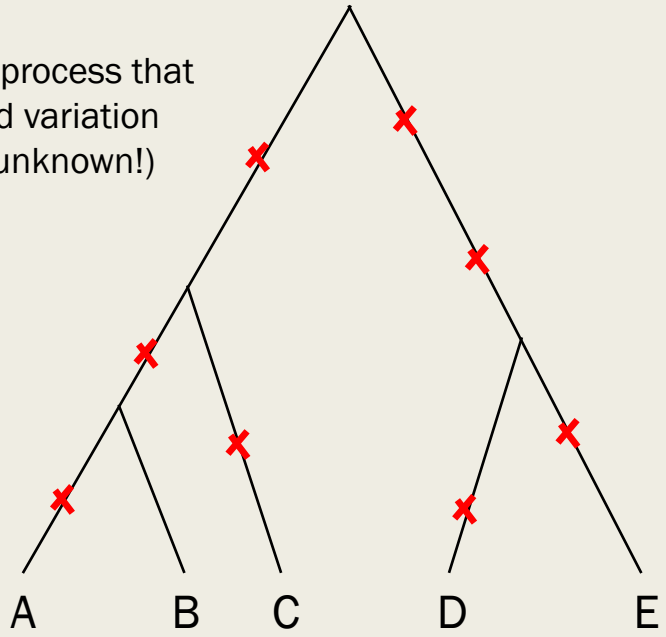


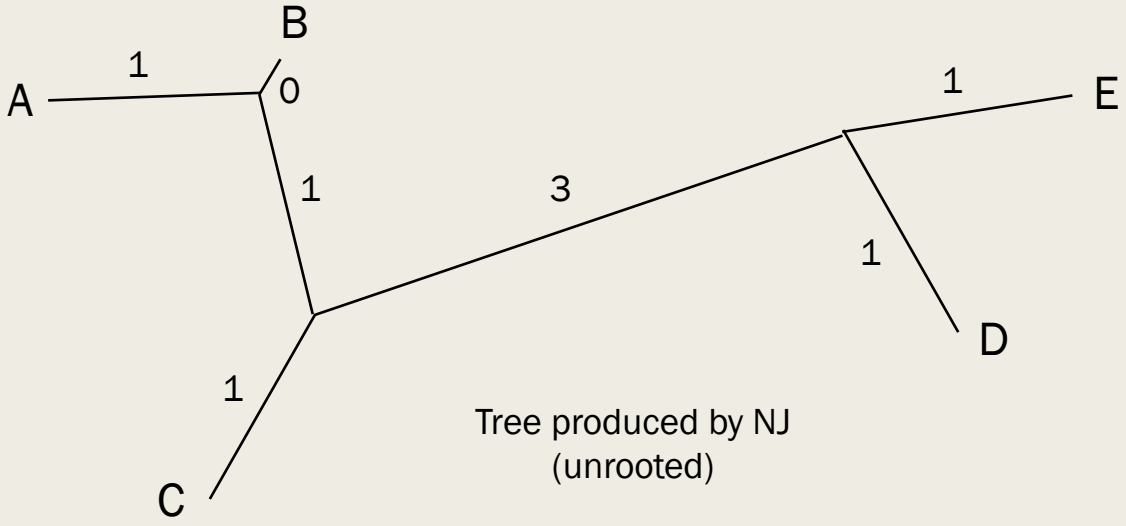
Original input dissimilarity map

δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

What have we learned from our example?

Mutational process that generated variation (usually unknown!)





Tree produced by NJ (unrooted)

Original input dissimilarity map

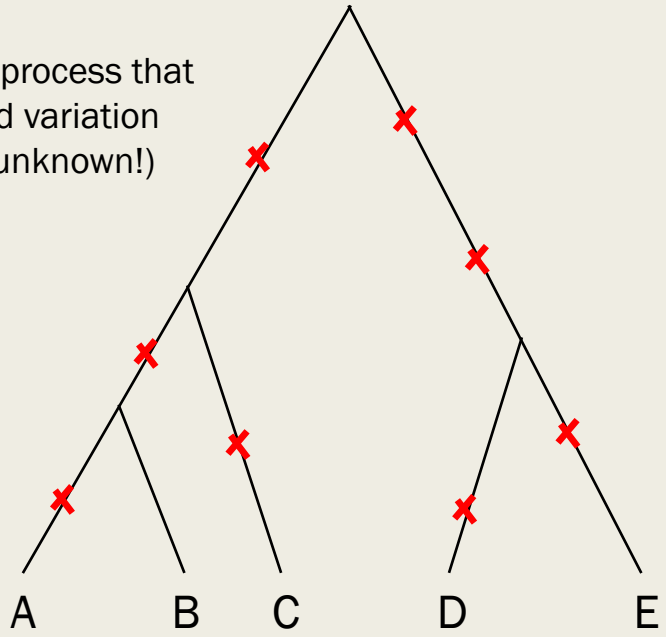
δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

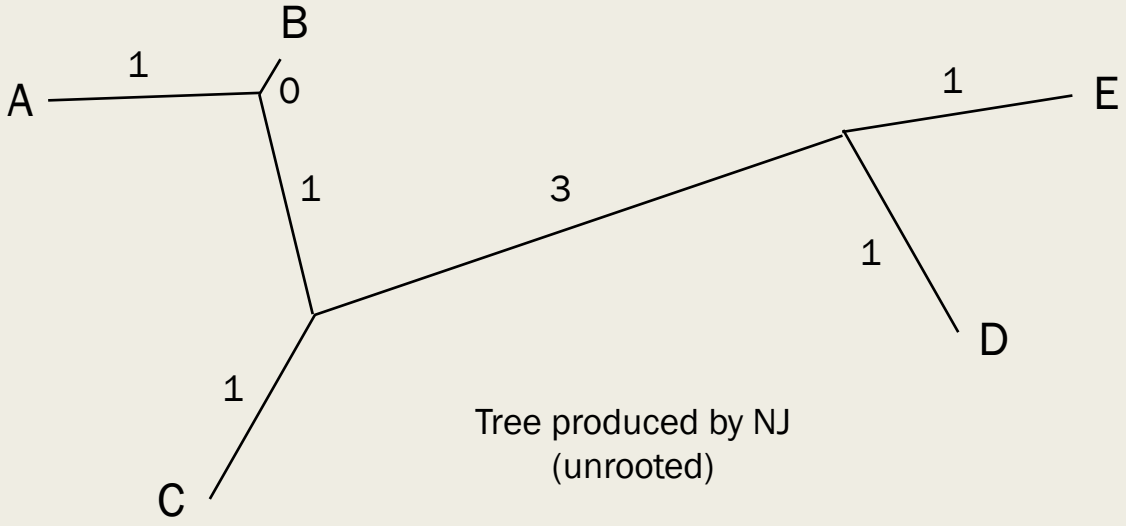
δ_{NJ}	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

Tree metric on *X* induced by NJ

What have we learned from our example?

Mutational process that generated variation (usually unknown!)





Tree produced by NJ (unrooted)

Original input dissimilarity map

δ	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

=

δ_{NJ}	A	B	C	D	E
A	0	1	3	6	6
B		0	2	5	5
C			0	5	5
D				0	2
E					0

Tree metric on *X* induced by NJ

Neighbor-Joining is consistent

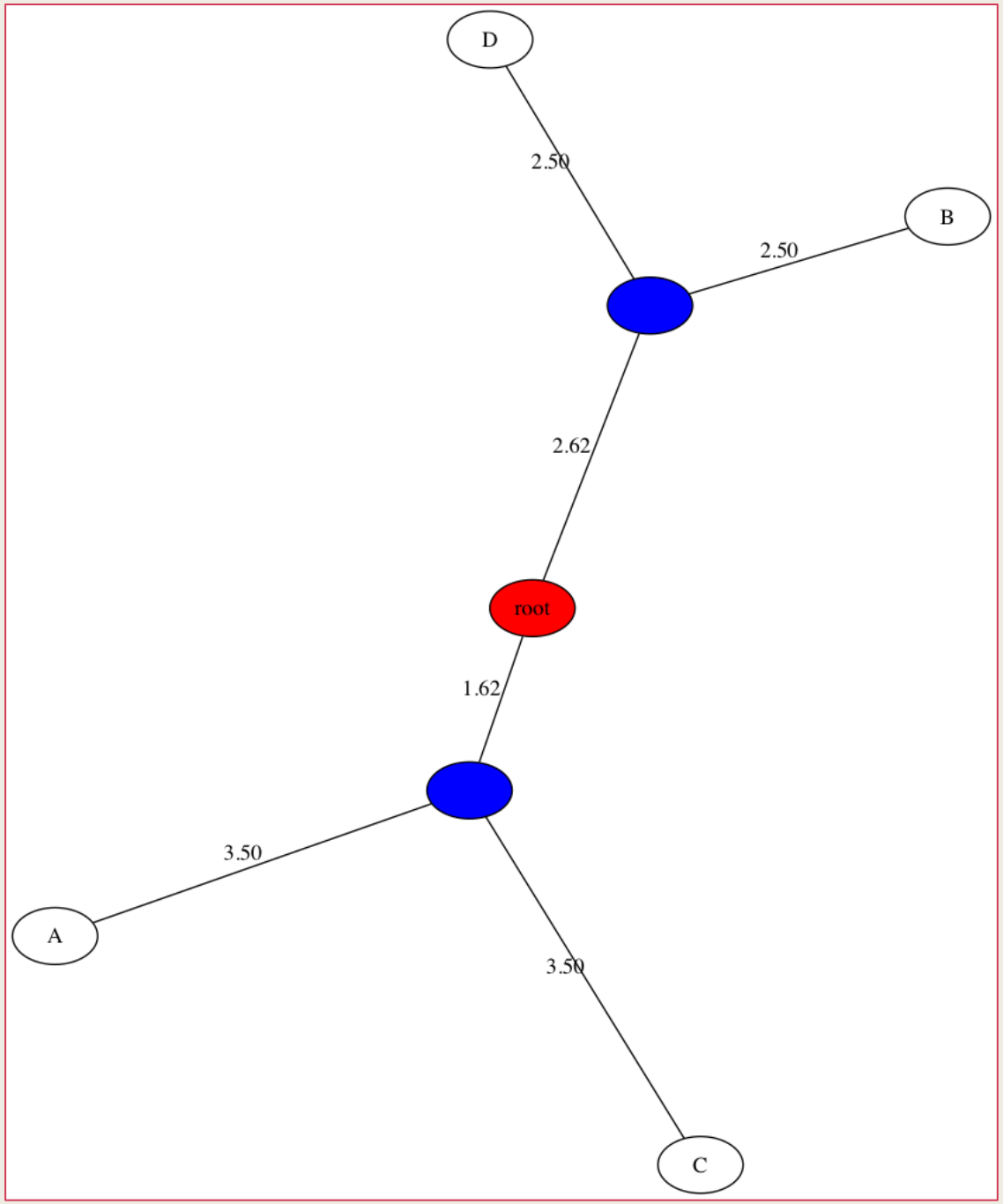
- If the original dissimilarity map is a tree metric, NJ will produce an induced tree metric equal to the original (***consistency***)

- UPGMA is not always consistent

- If the original dissimilarity map is not a tree metric (almost always the case), NJ will get closer, but both UPGMA and NJ are heuristics and not guaranteed to produce the edge-weighted tree that induces the very closest map to the original input (NP-complete)

UPGMA on Handout 15 example

δ	A	B	C	D
A	0	10	7	12
B		0	9	5
C			0	10
D				0

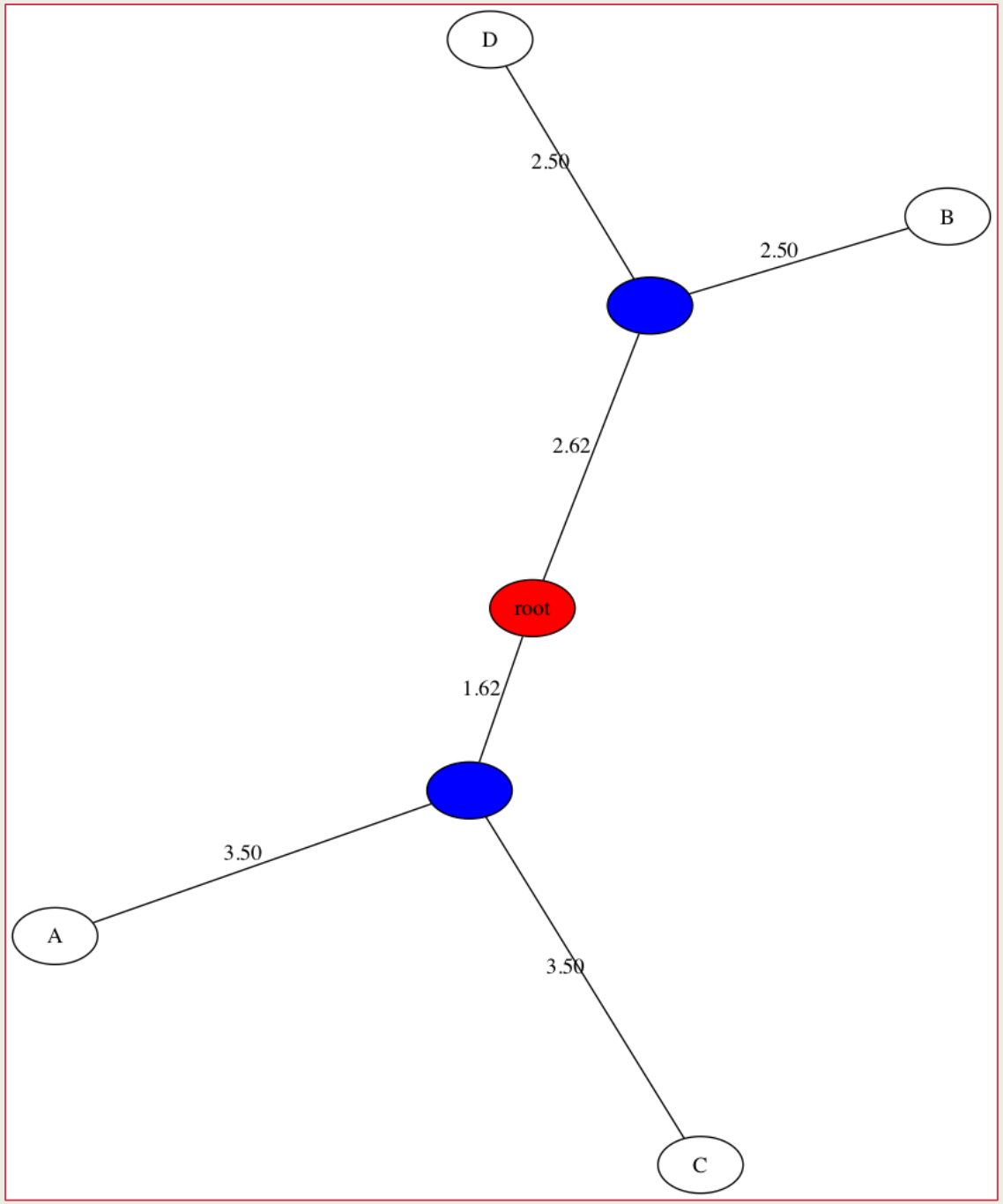


UPGMA on Handout 15 example

δ	A	B	C	D
A	0	10	7	12
B		0	9	5
C			0	10
D				0

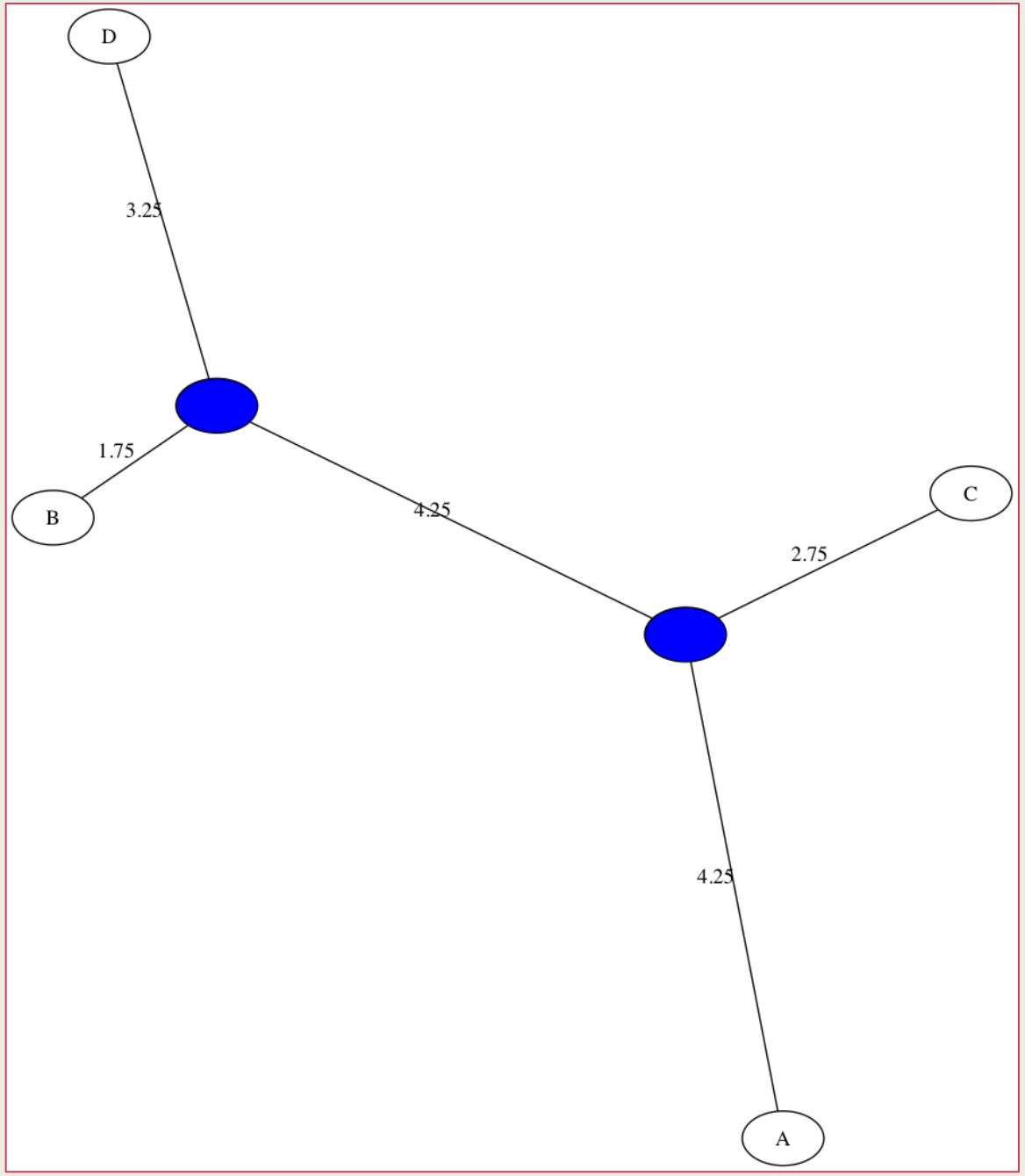
10.25 = (10+12+9+10)/4 (unweighted average)

δ_{UPGMA}	A	B	C	D
A	0	10.25	7	10.25
B		0	10.25	5
C			0	10.25
D				0



NJ on Handout 15 example

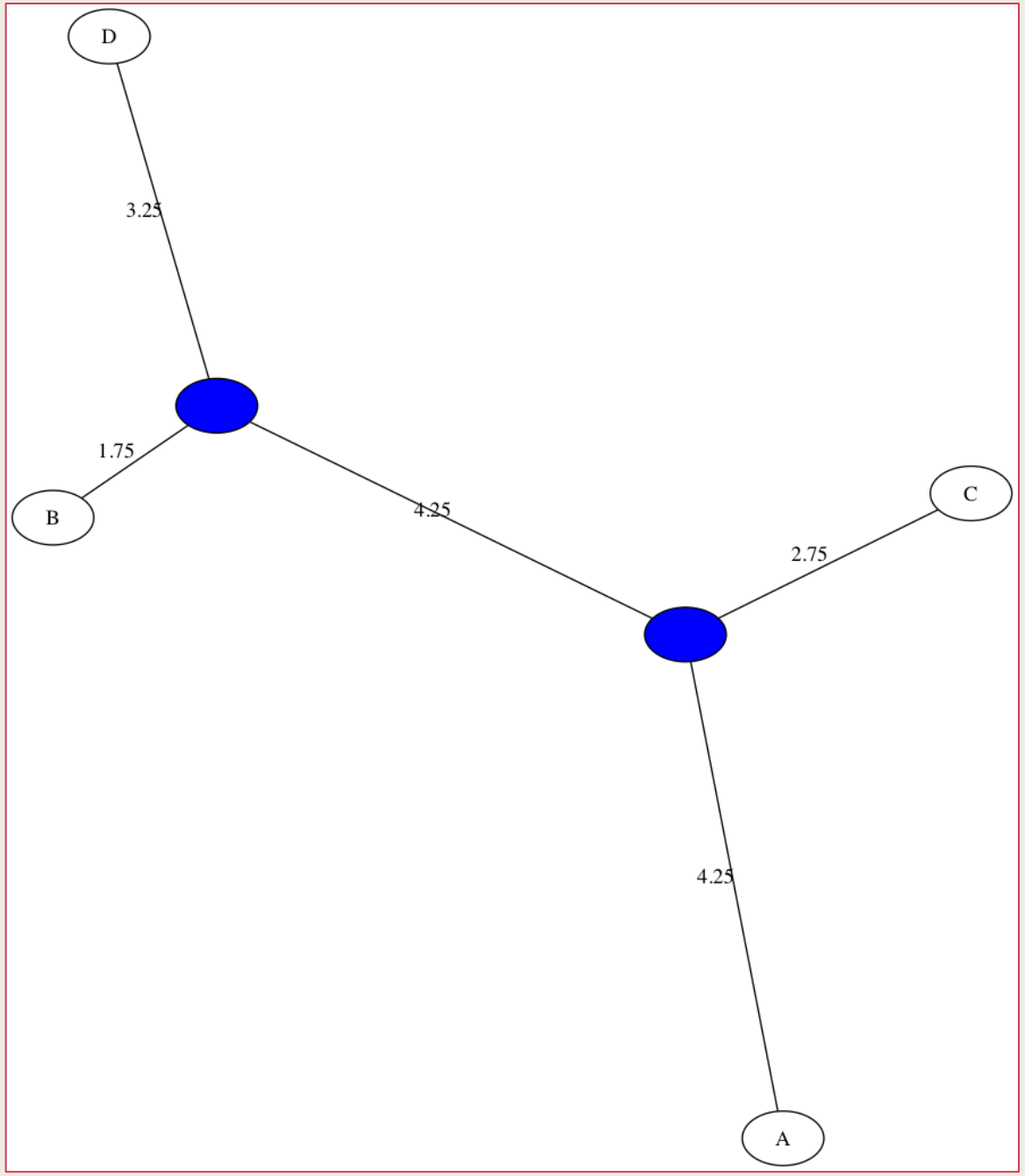
δ	A	B	C	D
A	0	10	7	12
B		0	9	5
C			0	10
D				0



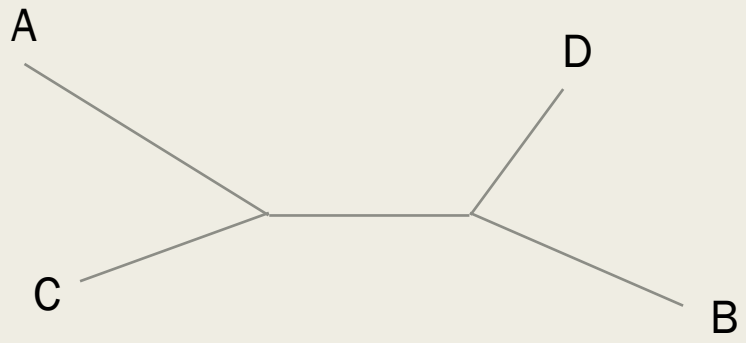
NJ on Handout 15 example

δ	A	B	C	D
A	0	10	7	12
B		0	9	5
C			0	10
D				0

δ_{NJ}	A	B	C	D
A	0	10.25	7	11.75
B		0	8.75	5
C			0	10.25
D				0

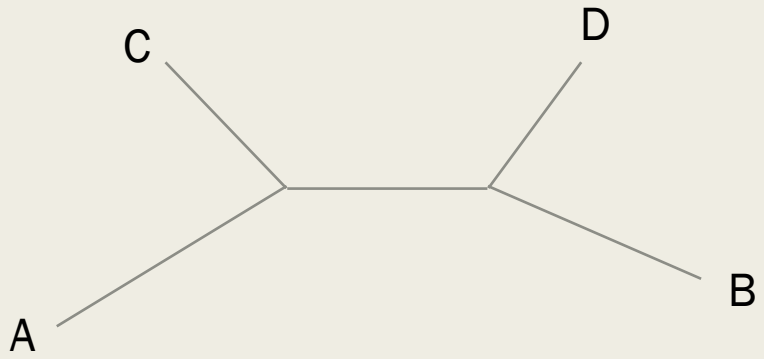


Handout 15 example



d(A,D)+d(D,B)+d(B,C)+d(C,A) = 12+5+9+7 = 33

δ	A	B	C	D
A	0	10	7	12
B		0	9	5
C			0	10
D				0



d(A,C)+d(C,D)+d(D,B)+d(B,A) = 7+10+5+10 = 32

Different ways of “walking” around the entire tree produce different lengths

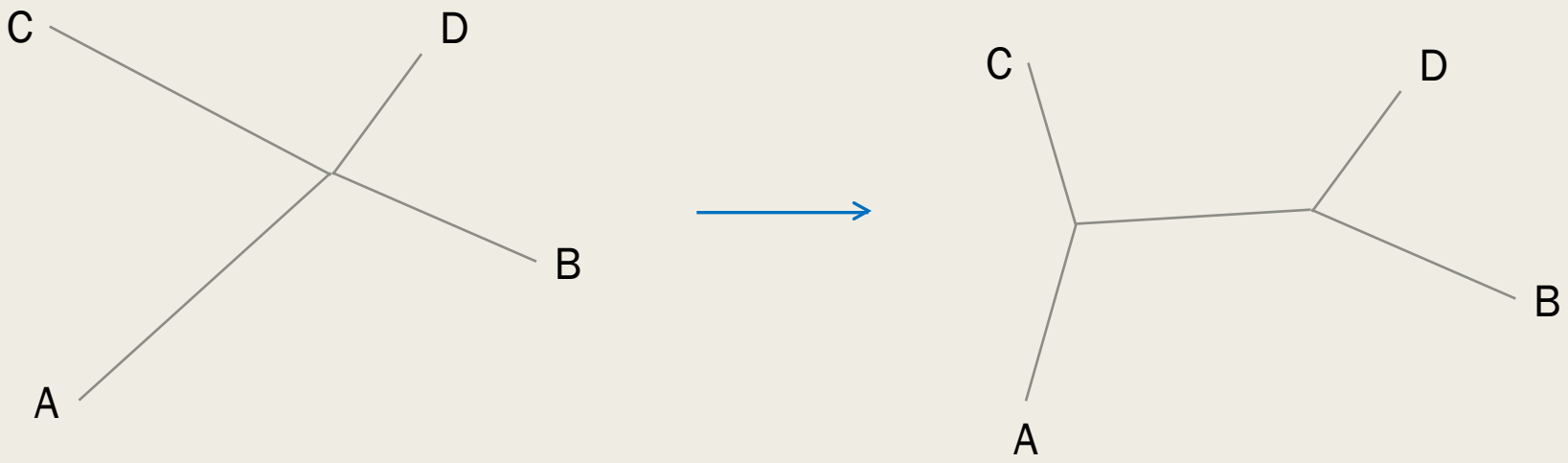
Q-criteria seeks to choose the neighbors that would minimize the average tree length the most

Q-criteria intuition

- Goal: we want the smallest tree that adequately explains the observed patterns of evolution (called BME: Balanced Minimum Evolution)

- Q-criteria minimizes the “whole tree length”, which is the average of all the different ways we could walk around the tree

- The idea is that we want to merge nodes that are far away, so we don’t have to “walk” to each of them separately, we can use the path to their merged vertex

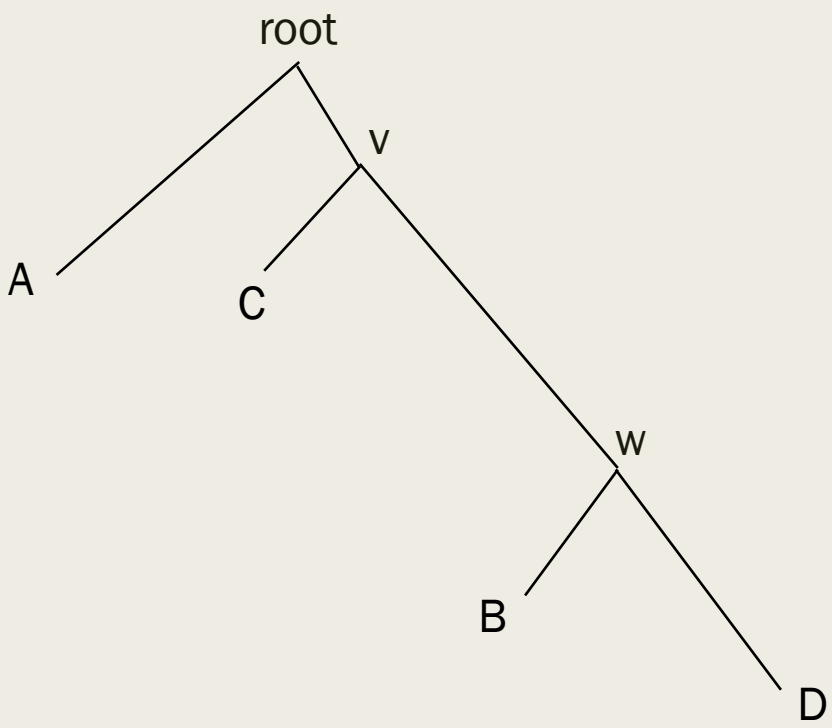


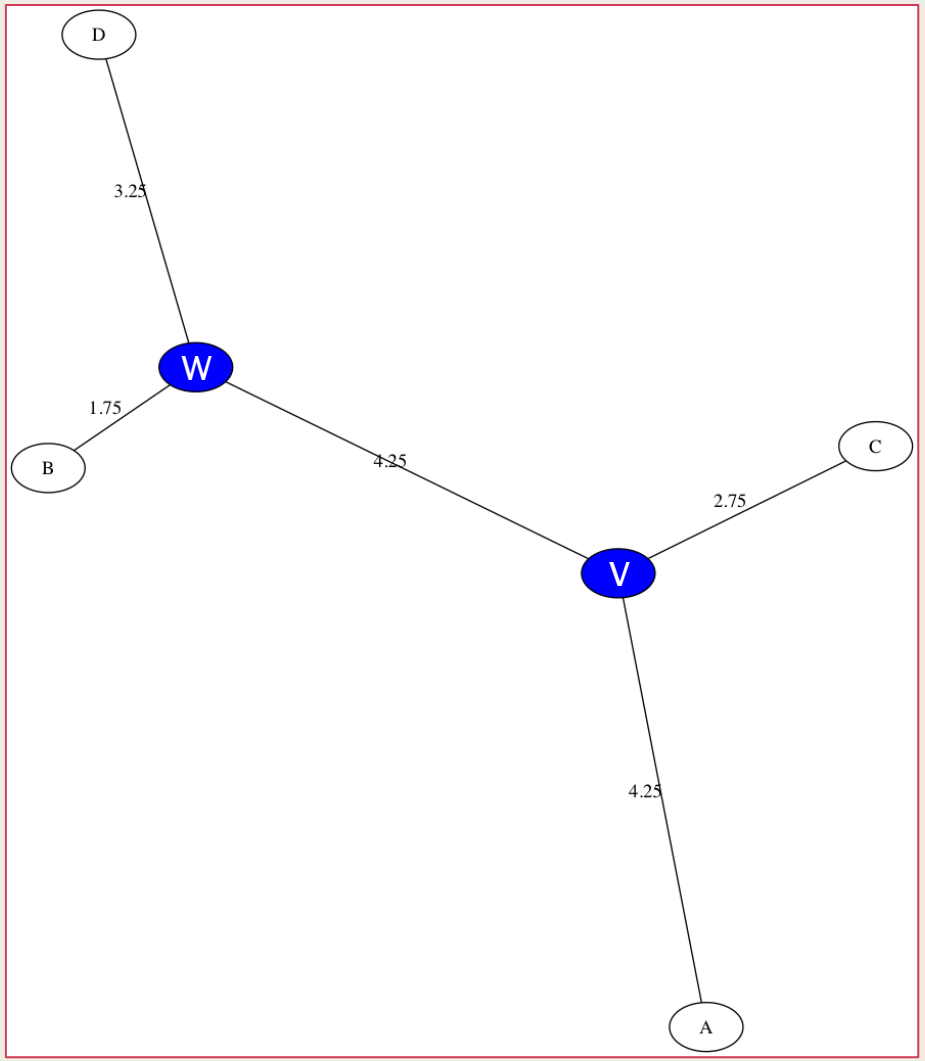
How to root a NJ tree?

- Method 1: use an ***outgroup***
- An outgroup is a species or sample that is more distantly related to all the other samples (“ingroup”) than any pair of ingroup samples

How to root a NJ tree?

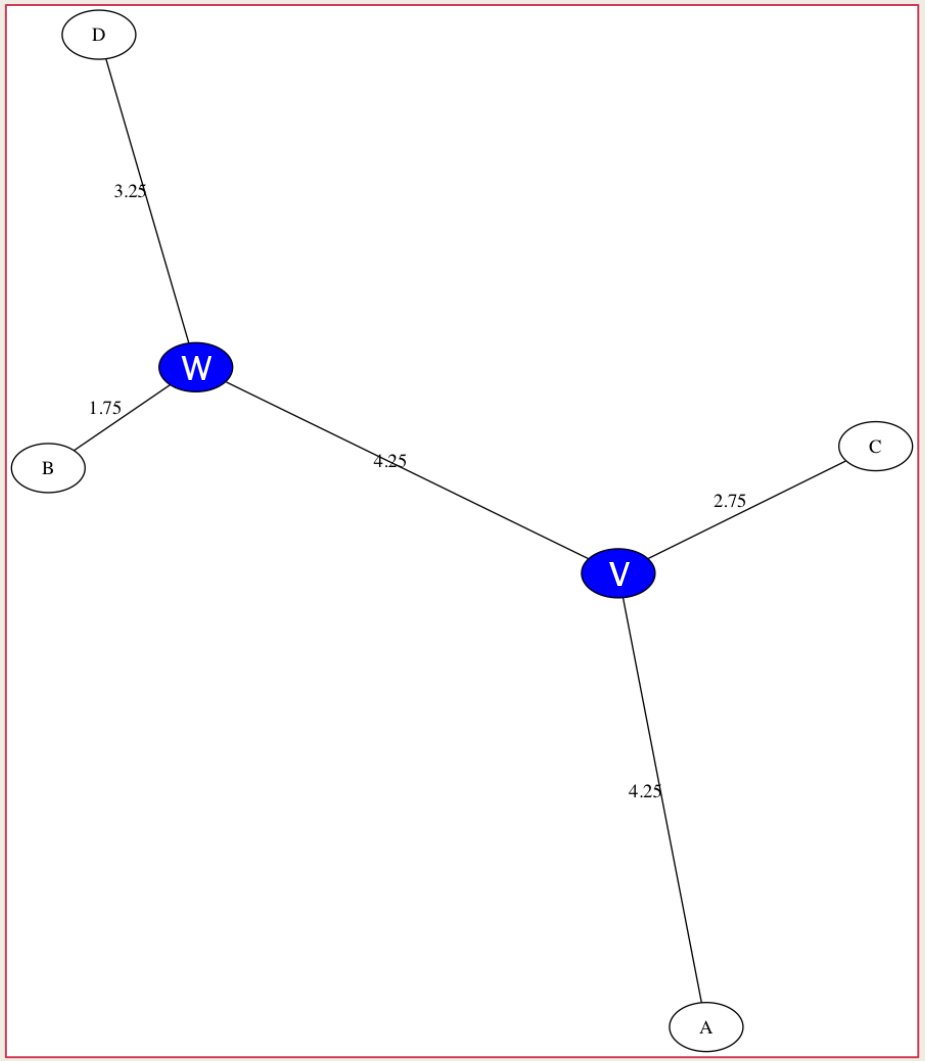
- Method 1: use an ***outgroup***
- An outgroup is a species or sample that is more distantly related to all the other samples ("ingroup") than any pair of ingroup samples
- For example, if we knew that A is an outgroup to ingroup {B,C,D}, we could root the NJ like this:





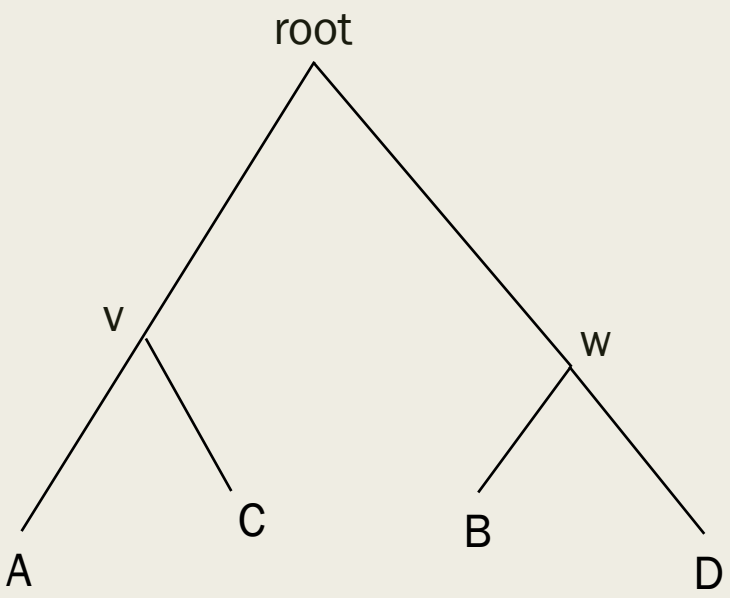
How to root a NJ tree?

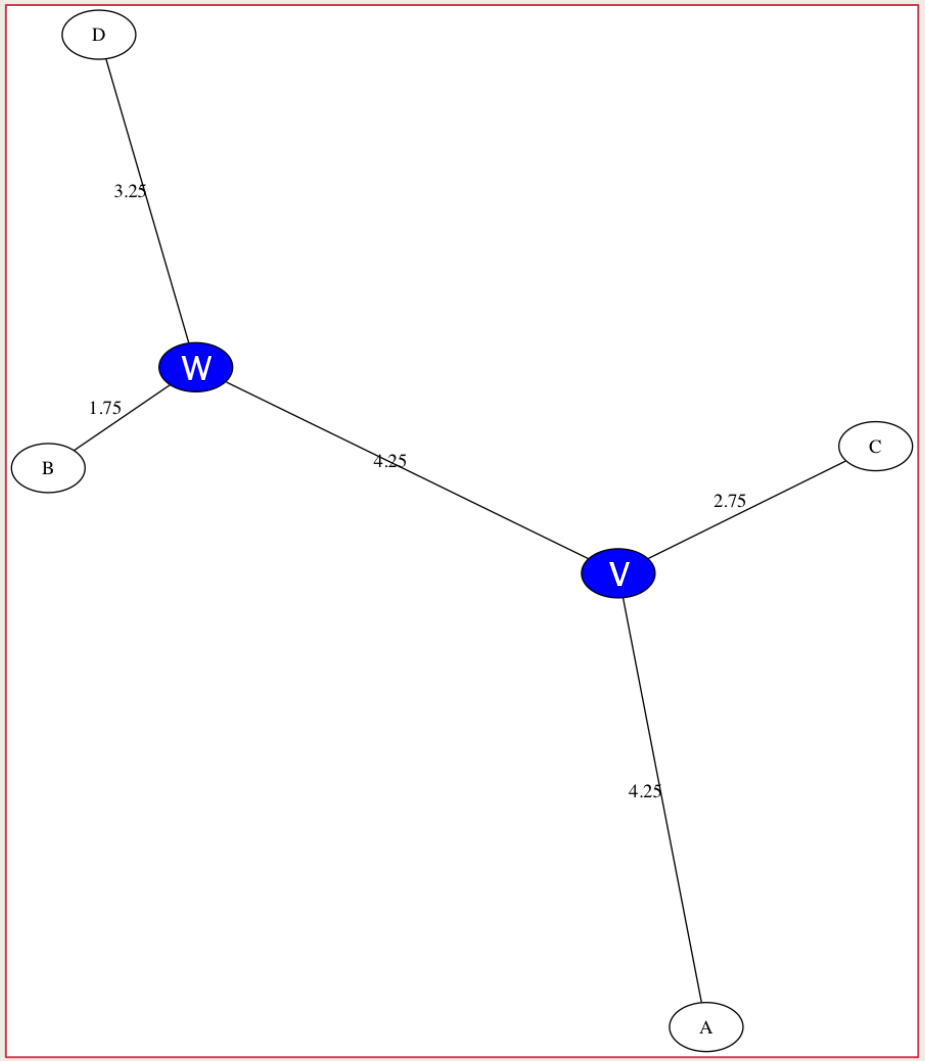
- Method 2: divide the longest path between leaves by 2
- *Assumption: molecular clock more or less valid*



How to root a NJ tree?

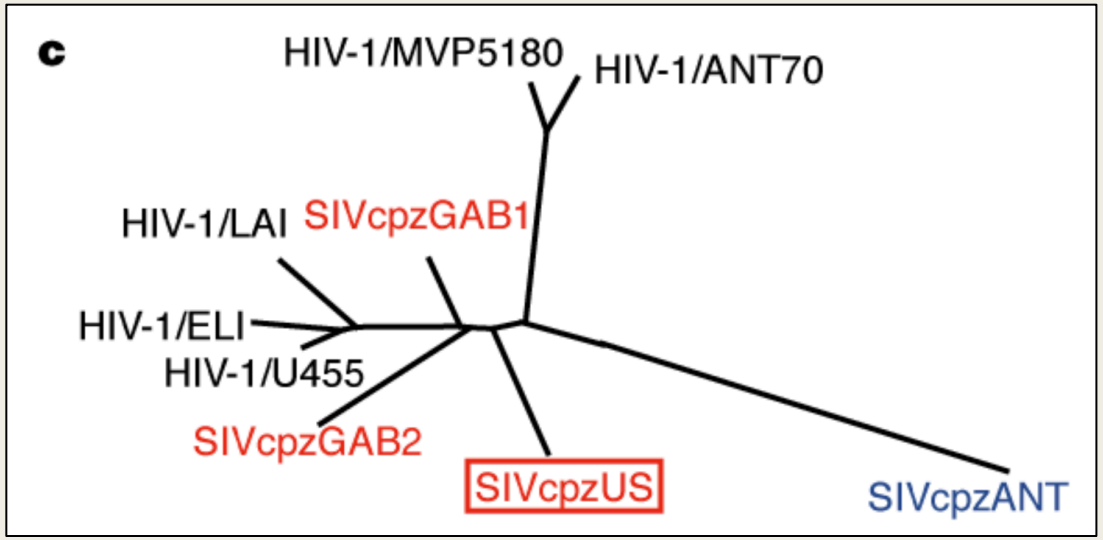
- Method 2: divide the longest path between leaves by 2
- *Assumption: molecular clock more or less valid*
- Longest path:
- A -> v -> w -> D = 4.25+4.25+3.25 = 11.75

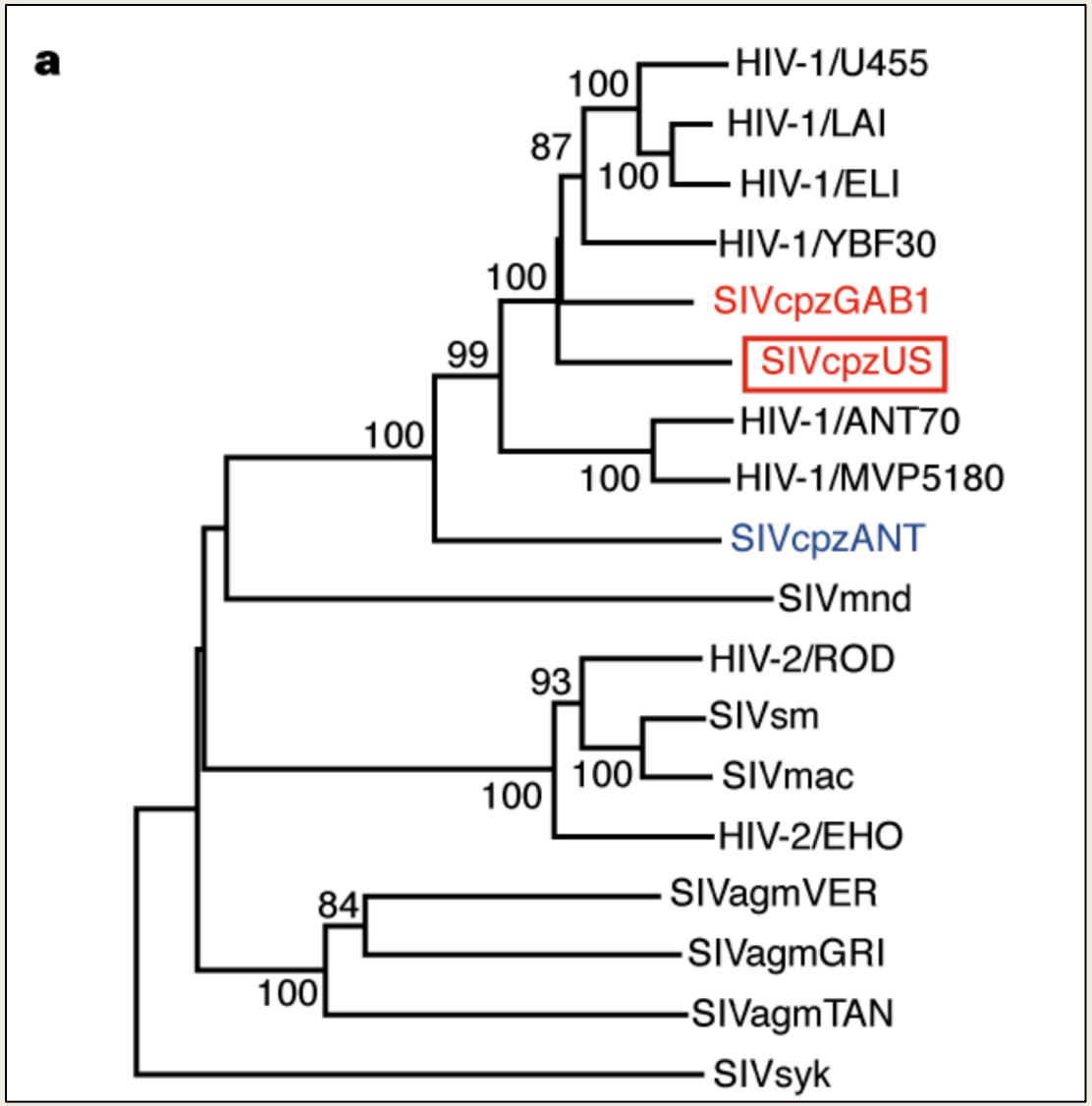




Example of NJ in research

Neighbor Joining trees (unrooted and rooted) for different strains of HIV





Origin of HIV-1 in the chimpanzee *Pan troglodytes troglodytes* (Nature, 2009)

Next topic: parsimony
(ancestral reconstruction)

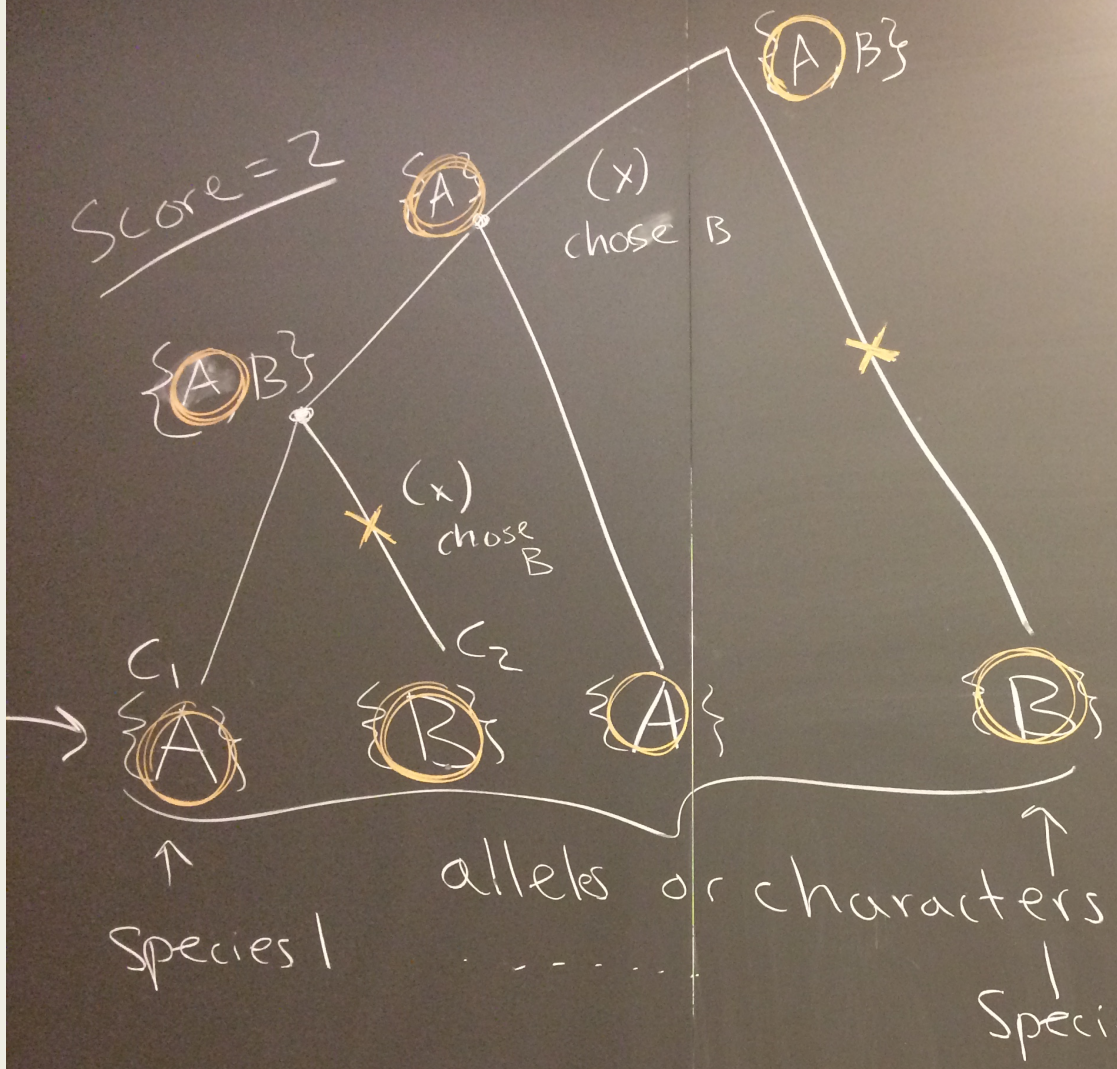
Parsimony Algorithms

- Ancestral Reconstruction

Input: tree + leaf labels.

↳ consider a single SNP

$S_1 = \{A\}$



Output

- character/allele at all internal nodes.
- Score for the tree = # of mutations

Fitch (1971)

bottom-up phase

S_v = state set of vertex v

$S_v = \{x\}$ if v is a leaf assigned state x

internal vertex v with children c_1, c_2

intersection

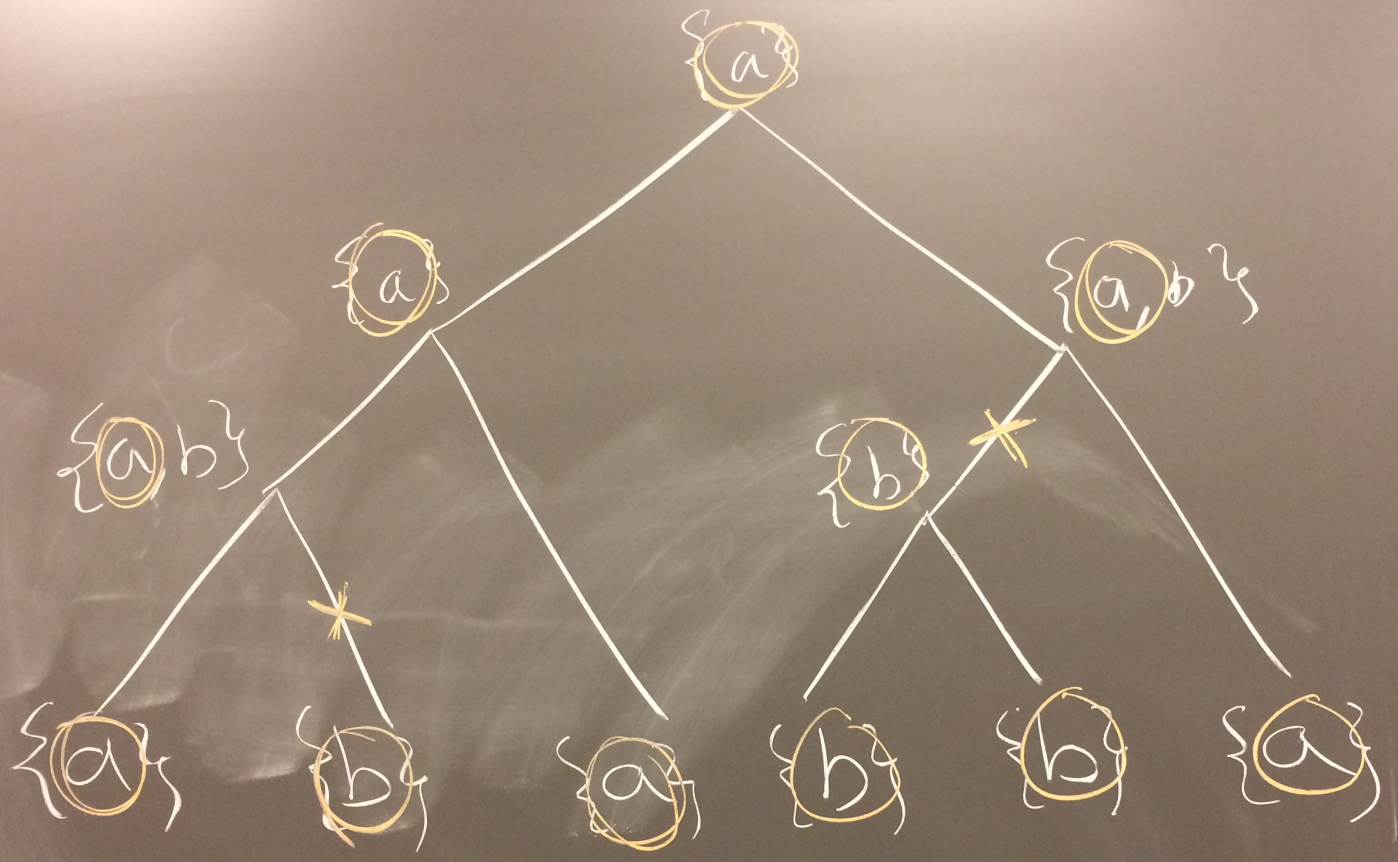
$$S_v = \begin{cases} S_{c_1} \cap S_{c_2} & \text{if } S_{c_1} \cap S_{c_2} \\ S_{c_1} \cup S_{c_2} & \text{o.w.} \end{cases}$$

union

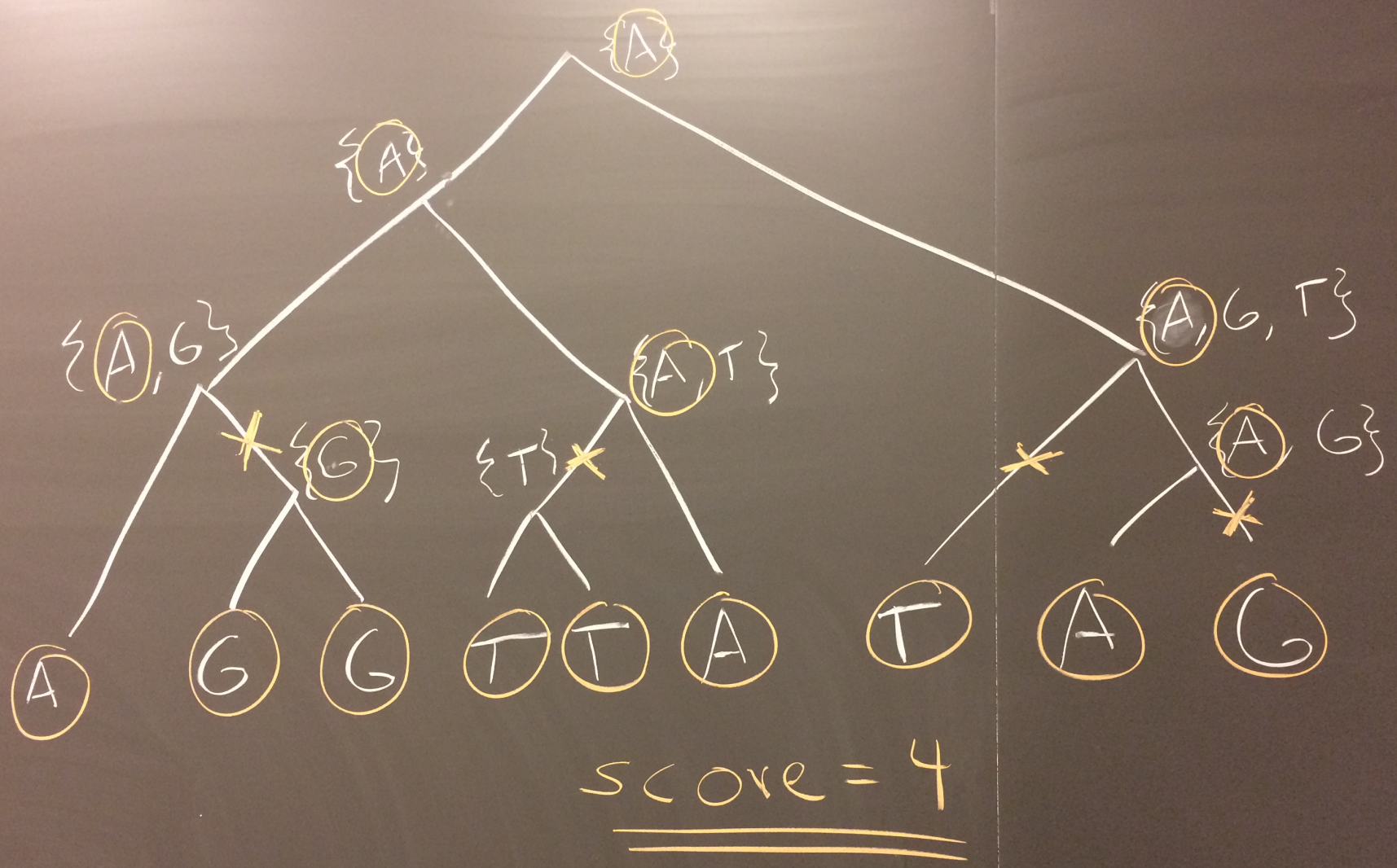
$\neq \emptyset$ empty set

top-down phase

- assign root arbitrarily
- assign node c_i
  - if parent of c is x and $x \in S_c$, then assign x to c
  - o.w. choose arbitrarily & add 1 to score



Score = 2



score = 4