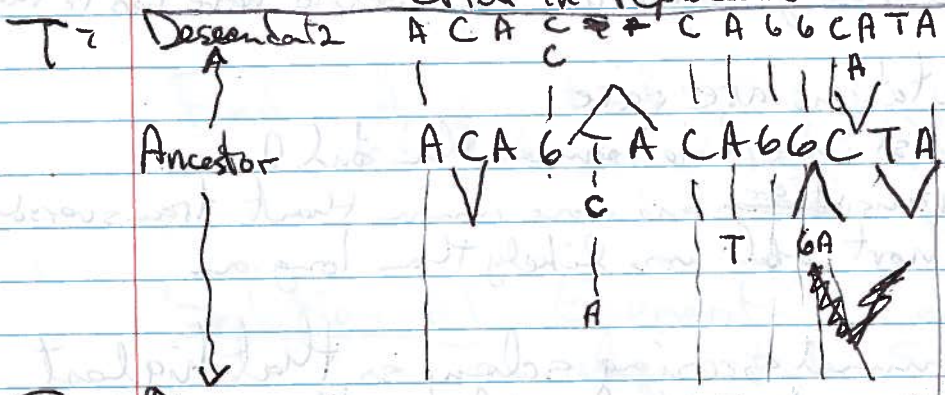
Sept 12 2016

Sequence alignment

Mutation: Change in sequence due to error in replication or unrepaired DNA damage



Substitution: Change one nuc for another

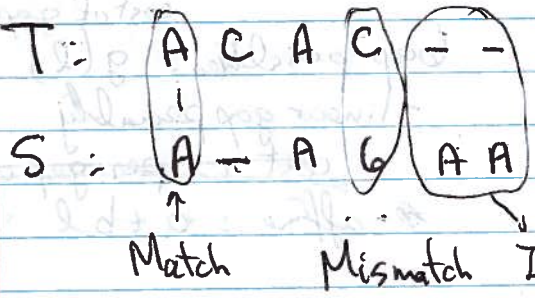
A ↔ 6 } transition
C ↔ T } common

A ↔ C, 6, T transversion (~3 times less common)

Insertion: Insert 1 or more nuc.

Deletion: Delete one or more

Alignment of S, T is obtained by inserting gaps in S and T to obtain S' and T', so that in S' and T', nucleotides derived from same ancestor occur at same position. Gaps reflect nucleotides missing due to del. or insertion



Problem: Ancestor and mutation scenarios are generally not known.

Idealized Pairwise Global Alignment Problem

Given: S, T
Find: Alignment of S and T that is most likely to reflect the evolutionary scenario that would have led to them

Key idea:
1. Mutations are rare
2. Subst. are more common than indel
3. Transitions are more common than transversions
4. Short indel more likely than long ones

⇒ Define alignment scoring scheme so that highest scoring alignment is the one that is the most likely to be correct.

Scoring a given alignment

```
S: A C A G T C - - T A
T: A T A - T A A A T A
   1 -1 1 -2 1 -1 -2 -2 1 1 : -3
```

Score: Subst. score + indel score

Subst. fct. or cost matrix

$$M = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{array}{c} \begin{array}{cccc} A & C & G & T \end{array} \\ \begin{pmatrix} +1 & -2 & -1 & -2 \\ -2 & +1 & -2 & -1 \\ & & \ddots & \\ & & & \ddots \end{pmatrix} \end{array}$$

Gap penalties: $g(\ell)$ = cost of gap of length ℓ

- linear gap penalty: $g\ell = -2\ell$ ~~cost c per gap character~~
- affine: $a + b\ell = -5 - 1\ell$

Pairwise seq. Aln. Problem (linear gap penalty)

Given: $S = S_1 \ldots S_m$, $T = T_1 \ldots T_n$ sequences
M subst cost matrix
c

Find: Alignment of S, T
s.t that score(Aln(S,T)) is maximized

Solution #1: Enumerate and evaluate all possible alignments for S and T, report best

Problem: Way too slow $\Omega(2^{m+n})$

Solution #2: Needleman-Wunsch algo. (1970)

S = AGCT
T = ACA

Dynamic Prog algo

$s_1\ s_2\ s_3\ s_4$
S = ACAT
T = AGT
$t_1\ t_2\ t_3$

X =

	·	A	G	T
·				
A				
C				
A				
T			○	

$X_{i,j}$ = Score of the best alignment for $s_1 \dots s_i$ against $t_1 \dots t_j$

Note: We want $X_{m,n}$, and we want the alignment that achieves this score.

How to calculate $X_{i,j}$?

Alignment	Score
Best Aln $\begin{pmatrix} s_1 \dots s_i \\ t_1 \dots t_j \end{pmatrix}$ either looks like Best Aln $\begin{pmatrix} s_1 \dots s_{i-1} & s_i \\ t_1 \dots t_{j-1} & t_j \end{pmatrix}$	$\rightarrow X_{i-1,j-1} + M($
OR Best Aln $\begin{pmatrix} s_1 \dots s_{i-1} & s_i \\ t_1 \dots t_j & - \end{pmatrix}$	$\rightarrow X_{i-1,j} + c$
OR Best Aln $\begin{pmatrix} s_1 \dots s_i & - \\ t_1 \dots t_{j-1} & t_j \end{pmatrix}$	$\rightarrow X_{i,j-1} + c$

So: $X_{i,j} = \max \begin{cases} X_{i-1,j-1} + M(s_i, t_j) \\ X_{i-1,j} + c \\ X_{i,j-1} + c \end{cases}$ (when $i>0, j>0$)

$X_{0,j} = j \cdot c$
$X_{i,0} = i \cdot c$
$X_{0,0} = 0$

	·	A	G	T
·	0	-2	-4	-6
A	-2	+1	0	-2
C	-4	-1	0	-1
A	-6	-3	-2	-3
T	-8	-5	-4	-1

$c = -2$

Calculate X entri
to right, top

For each entry, keep
to entry that achieved

$X_{4,3} = -1$: Score of best alignment

We recover best alignment following backpointers from (m,n)
↖ : Match
← : Gap in S
↑ : Gap in T

Best = A C A T
Alm = A - G T

Running time: $O(m \cdot n)$
Space: $O(m \cdot n)$

Note: NW algo can be modi
with affine gap pen
→ Why does it not al
→ What is the modific

To recover all optimal alignments, do Depht-First-Search
backpointers starting from (m,n).

Pairwise local alignment problem

Idea: Often, two sequences only share significant sim
over a particular region.
Examples: Two

Given: Sequences $S = s_1 \ldots s_m$ and $T = t_1 \ldots t_n$
Subst. matrix M
gap penalty c

Find: Indices i, j, k, ℓ, where $1 \le i \le j \le m$, $1 \le k \le \ell \le n$

such that $\text{Score}\left(\text{BestAln}\left(\begin{matrix} s_i \ldots s_j \\ t_k \ldots t_\ell \end{matrix}\right)\right)$ is maximized

Can we do better than this?

```
for i:1...m
    for j:i...m
        for k:1...n
            for l:k...n
                NW(s_i...s_j, t_k...t_l)
```

$O(m^3 n^3)$

Smith Waterman algo.

$X_{i,j}$ = Score of best local alignment for $s_1 \ldots s_i, t_1 \ldots t_j$ where s_i and t_j are included in alignment

$$X_{i,j} = \max \begin{cases} X_{i-1,j-1} + M(s_i, t_j) \\ X_{i-1,j} + c \\ X_{i,j-1} + c \\ 0 \end{cases}$$

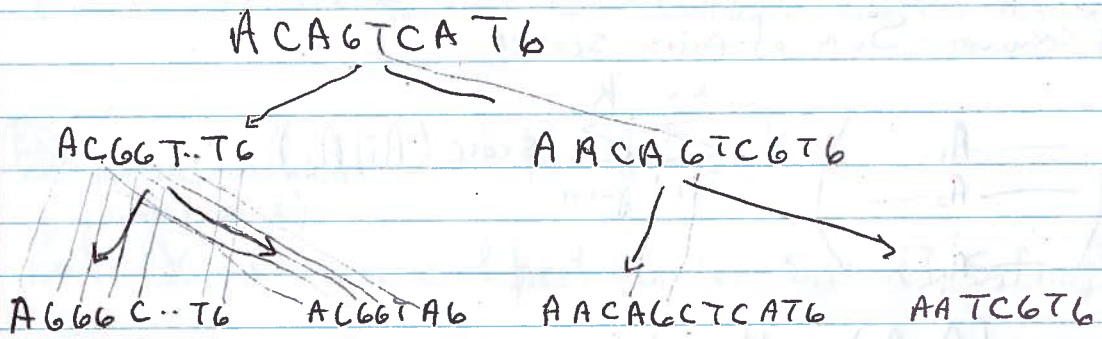
$X_{i,0} = 0 \quad \forall i = 1 \ldots m$
$X_{0,j} = 0 \quad \forall j = 1 \ldots n$

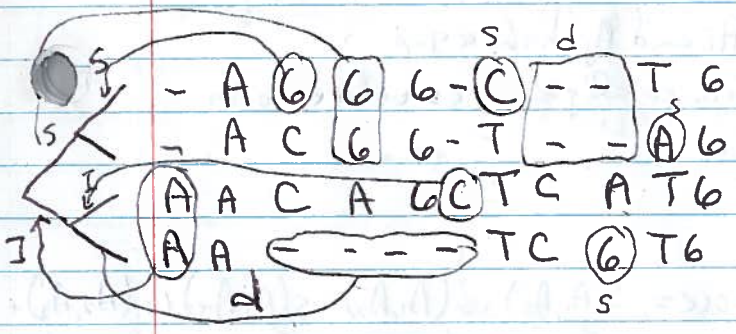
Traceback from $\underset{i \in \{1 \ldots m\}, j \in \{1 \ldots n\}}{\text{argmax}} \{X_{i,j}\}$ until hitting zero

4522 Pontiac
→ 3452

Notes: Local alignment problem only makes sense when $\sum_{a \in \Sigma, b \in \Sigma} M(a,b) < 0$

Multiple seq. alignment.





Biological problem: Given | n sequences
Phylogenetic tree T, where each sequence is assigned to a leaf

Find: Multiple alignment where all nucleotides derived from the same common ancestor are aligned to each other.

Scoring a given MSA

Goal: Find a scoring function $score: MSA \to \mathbb{R}$
such that $score(A) > score(A') \Rightarrow A$ is more likely to be biologically correct than A'

Many choices of scoring functions are possible
Most common: Sum-of-pairs score

$$Score\begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_k \end{pmatrix} = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} score(A_i, A_j)$$

where $score(A_i, A_j)$ is obtained by
1. Removing positions where both A_i and A_j have a gap
2. Scoring the remaining pairwise alignments as before, with subst. Matrix M, gap penalty c.

Example:

1. A C T
2. A 6 –
3. 6 T
4. – 6 T

$Score = s(A_1, A_2) + s(A_1, A_3) + s(A_1, A_4) + s(A_2, A_3) + s(A_2, A_4) + s(A_3, A_4)$

$= -3 - 6 - 6 - 3 - 3 + 2$

$= -19$

Optimal MSA problem

Given: Sequences $S_1 \ldots S_n$
Subst. matrix M
Gap penalty c

Find: MSA for $S_1 \ldots S_n$ that achieves the highest possible

Generalization of N-W algorithm
(For n=3)

Let $X_{i,j,k}$ = score of best aln for $S_1[1 \ldots i]$, $S_2[1 \ldots j]$, $S_3[1 \ldots$

$$X_{i,j,k} = \max \begin{cases} X_{i-1,j-1,k-1} + M(S_1[i], S_2[j]) + M(S_1[i], S_3[k]) + M(S_2[j], S \\ X_{i-1,j-1,k} + M[S_1[i], S_2[j]] + 2 \cdot c \\ \vdots \\ X_{i,j,k-1} + 2c \end{cases}$$

For 3 seq. of length l
Dyn Prog table has $l \times l \times l$ entries

For n seq of length l : $O(l^n \cdot 2^n)$

↓

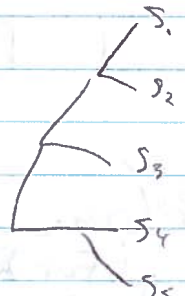
Exponential time algorithm

↗ Heuristic, with no guarantee of optimality

Progressive alignment Algo.

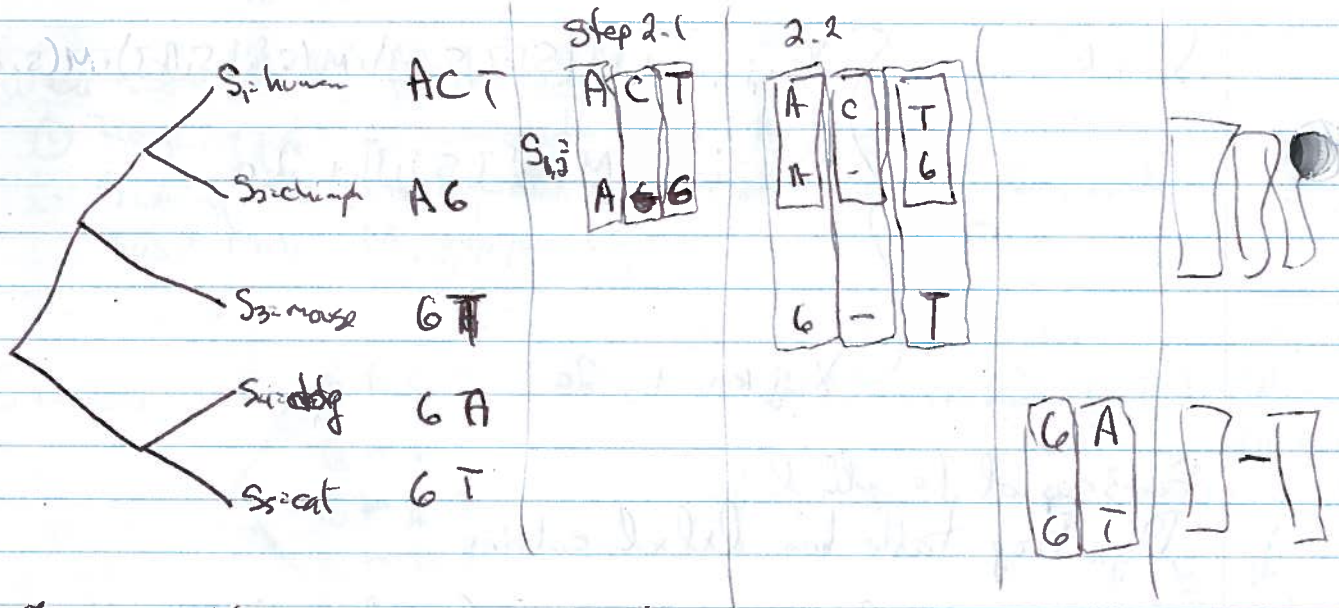
Input: $S_1 \dots S_n$, M, c

① Guess phylogenetic tree for $S_1 \dots S_n$



② For each internal node in ~~post~~ order traversal (from leaves to root)

- Find optimal alignment btw pairs of seq or alnt at two children

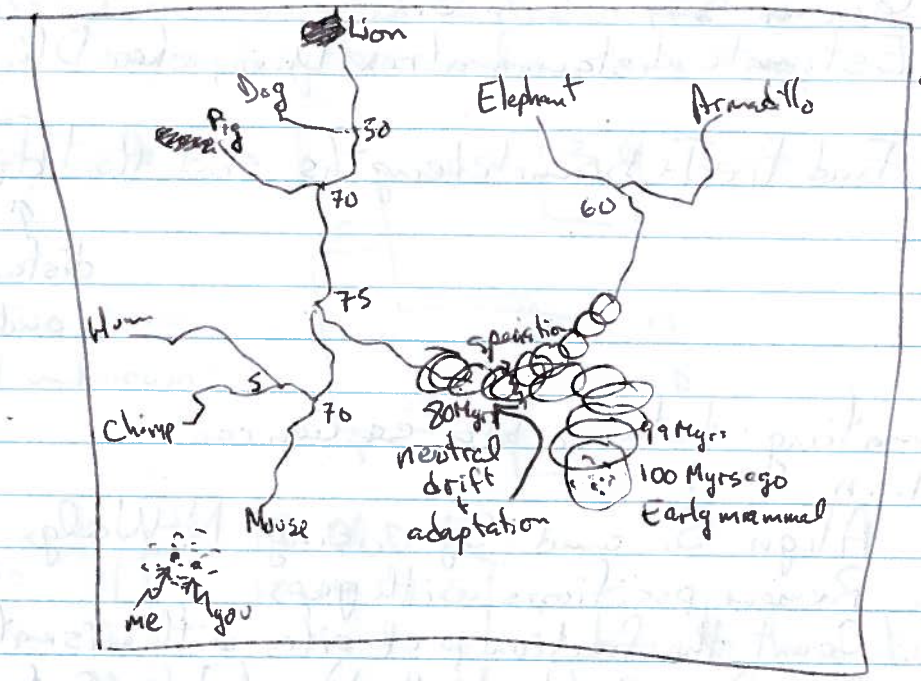


2.1. Align S_1 vs S_2 to obtain $S_{1,2}$

2.2 Align $S_{1,2}$ vs S_3 to obtain $S_{1,2,3}$

2.3 Align S_4 vs S_5 to obtain $S_{4,5}$

Sequence Evolution + Phylogenetics



$\{A, C, G, T\}^*$ = Space of all DNA sequences

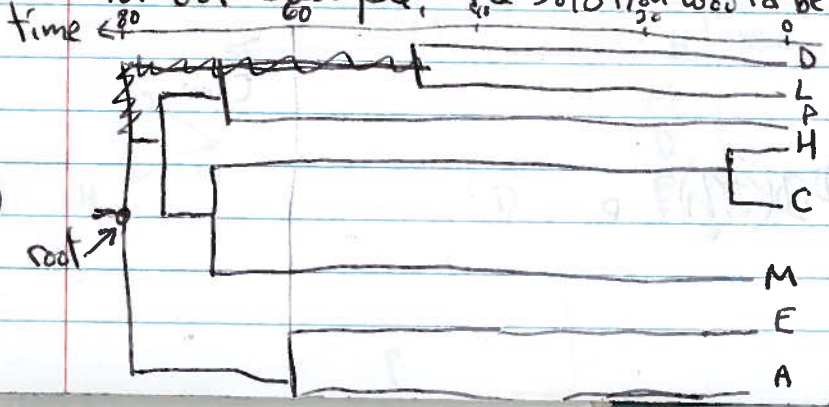
Phylogenetic Inference Problem (Biological version)

Given: DNA sequences from multiple species

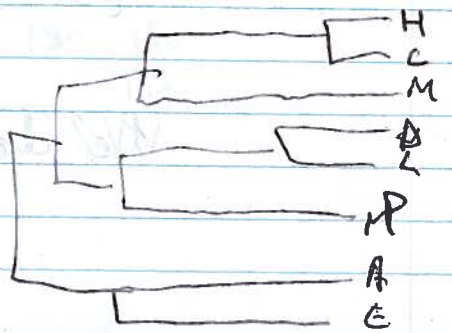
Find: Evolutionary tree that describes their evolution

→ Topology, Dating of speciation events

For our example, the solution would be:



Note: this is the same topology as



Distance-based Phylo. Inference methods

Idea: Given seq. $S_1 \ldots S_n$

(A) Estimate distance matrix $D_{n \times n}$, where $D(i,j)$ = distance btw S_i, S_j

(B) Find tree + Branch lengths such that $d_T(i,j) \approx D(i,j)$

($d_T(i,j)$: distance btw nodes i and j in tree T)

(A) Estimating distance btw sequences

For $i,j = 1 \ldots n$

1. Align S_i and S_j using N-W alg.
2. Remove positions with gaps
3. Count the fraction p of sites with mismatches
4. $D(i,j) = -\frac{3}{4}\left(\log\left(1-\frac{4}{3}p\right)\right)$ (Jukes-Cantor model)

Example

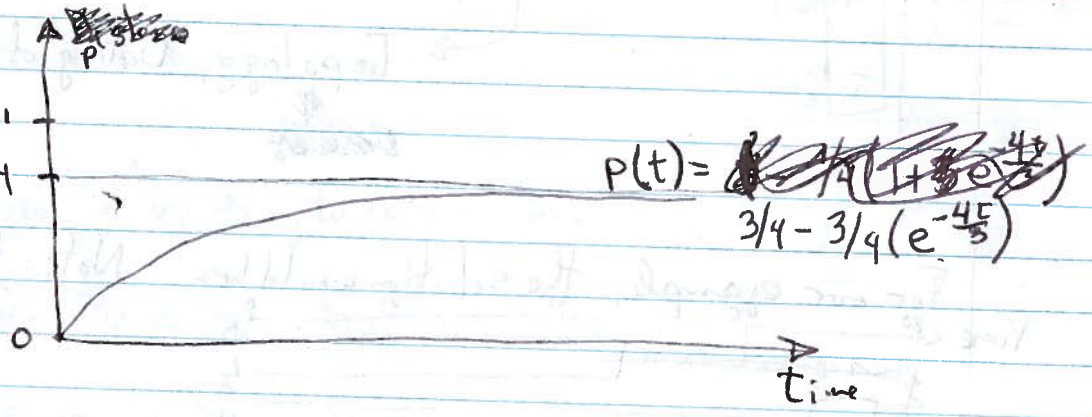
S_i: A C - - T G A C T G

S_j: A G T T T A - C A G

$\Rightarrow p = 3/7$

$D(i,j) = 0.63$

Note: If the two seqs. are very long



$p(t) = 3/4 - 3/4(e^{-\frac{4t}{3}})$

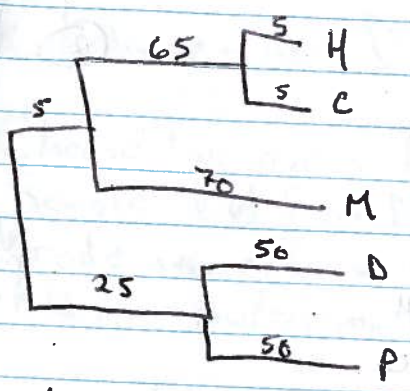
We choose $D(i,j) = p$

~~Under Kimura 2-parameter model with rates of transition vs transversion~~

Notes: More advanced distance measures consider separately transitions and transversions

Example:

True tree (unknown)



$D =$

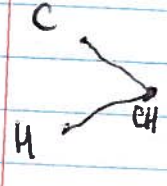
	H	C	M	D	P
H	-	10	140	150	150
C		-	140	150	150
M			-	150	150
D				-	100
P					-

How to find tree?

UPGMA algorithm

Repeat

1. Choose two closest nodes u, v according to $\arg\min_{i,j} D(i,j)$ ← UPGMA rule
2. Merge two nodes into a new node w
3. Remove rows/cols u, v from matrix
4. Insert new row/col for w : $D(w,i) = D(u,i) - \frac{1}{2} D(u,v)$



	CH	M	D	P
CH	~~135~~	185	145	145
M		-	150	150
D				100
P				

M

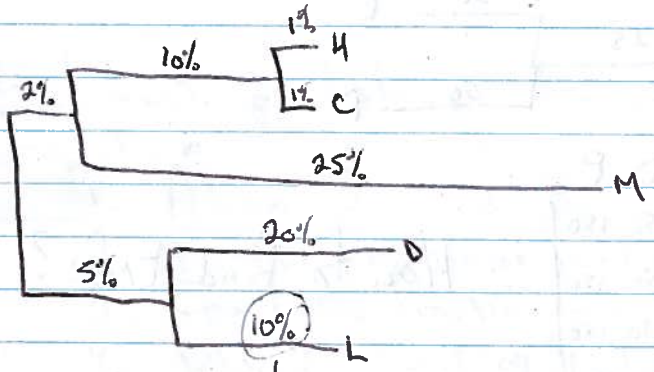
D

P

Running time: ?

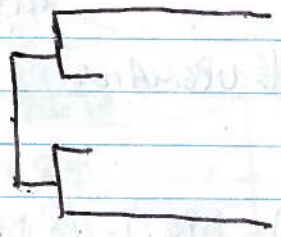
Problem: Only works if ① Distances are estimated perfectly accurately
② Mutation rate is constant

More realistic tree:



Mutation rate varies
⇒ Distances btw seq cannot be calculated in Mys but instead in Expected substitution per site

This means that 10% positions would have gotten mutated. the expected # of sub/s. per site is 0.1



⇒ UPGMA will produce the wrong tree

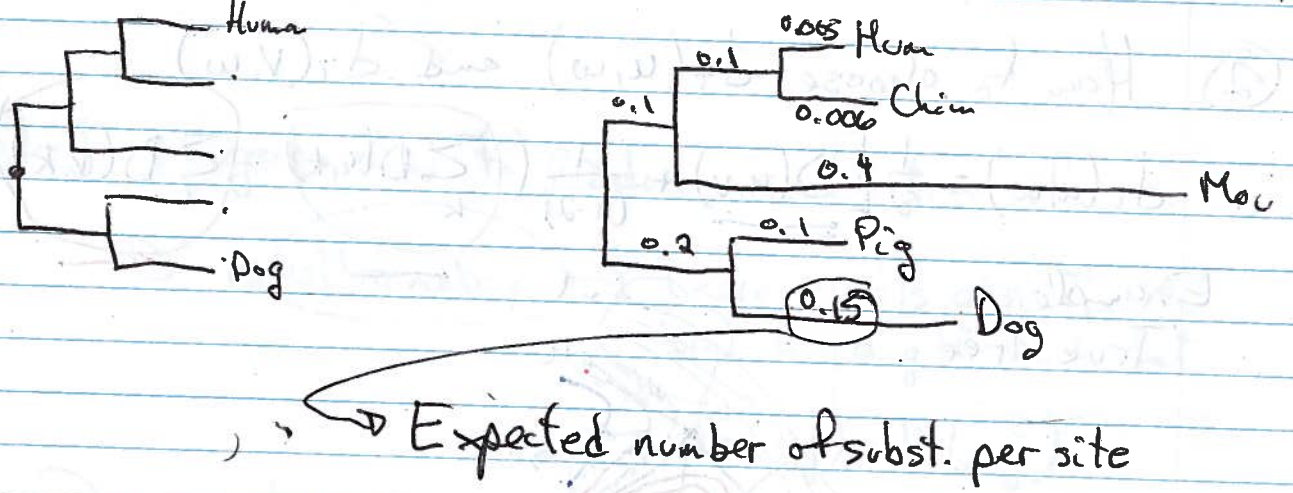
Unweighted Pair-Group Method with Arithmetic Mean

Reminder: UPGMA Algo

Input: $S_1 \ldots S_n$
Output: Tree T with branch lengths

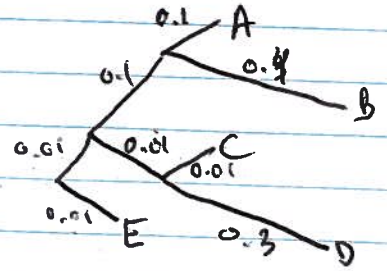
1. Estimate distance matrix D from pairwise alignments
2. Repeat
   - 2.1. Choose two nodes (u,v) to pair up
   - 2.2. Remove (u,v) from D
   - 2.3. Create new node w as ancestor of u,v
   - 2.4. Add new row/column for w in D.

UPGMA produces correct output if
1. Distances are estimated perfectly accurately
2. Mutation rate is constant

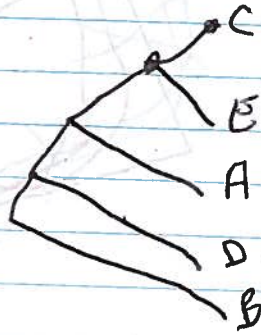


→ Expected number of subst. per site

Suppose true tree



UPGMA



Wrong Tree

Neighbor-joining Algo (Nei, Saitou)

Same as UPGMA, but different node pair selection rule

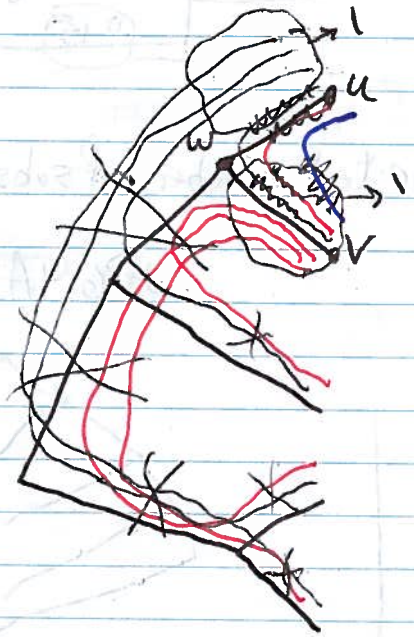
① Calculate $Q_{n\times n}$, where $Q(i,j) = \sum_{k=1}^{n} D(i,k) + \sum_{k=1}^{n} D(j,k) - (n-2)D($

Theorem: $u, v \leftarrow \underset{i,j}{\arg\max}\{Q(i,j)\}$, then nodes u and v must be a cherry

② How to choose $d_T(u,w)$ and $d_T(v,w)$

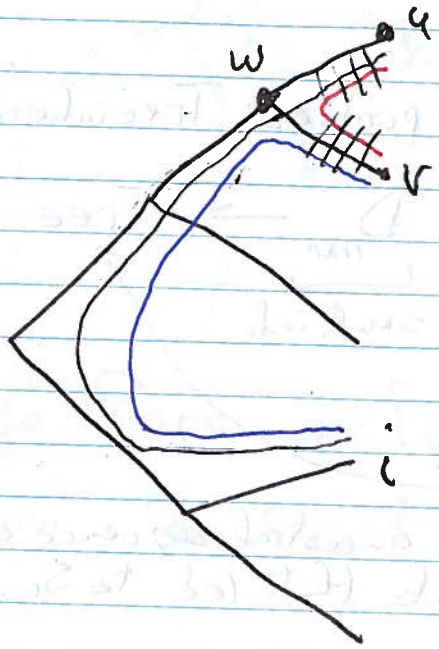
$$d_T(u,w) = \frac{1}{2}\left[D(u,v) + \frac{1}{(n-2)}\left(\sum_k D(u,k) - \sum_k D(v,k)\right)\right]$$

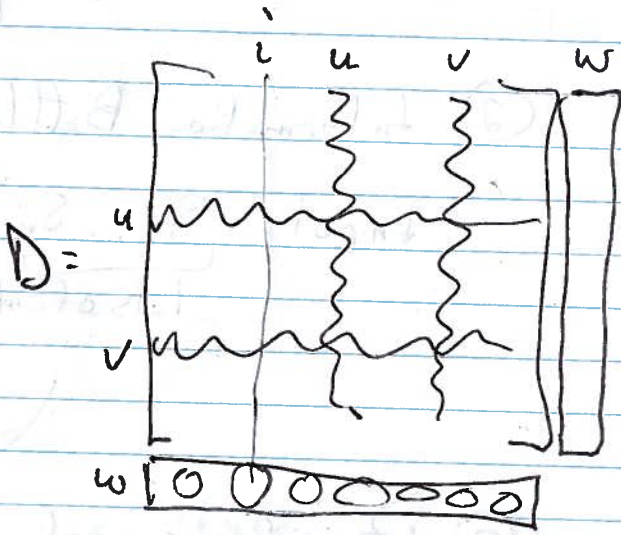
Example:
True Tree



Adding row/col for w in D

$$D(w,i) = \frac{1}{2}\left(D(u,i) + D(v,i) - D(u,v)\right)$$





N-J is guaranteed to produce correct tree if

D is ultrametric → There exists a tree with branch length such that

$$d_T(i,j) = D(i,j)$$

Running time: $O(n^4)$, but can be reduced to $O(n^3)$

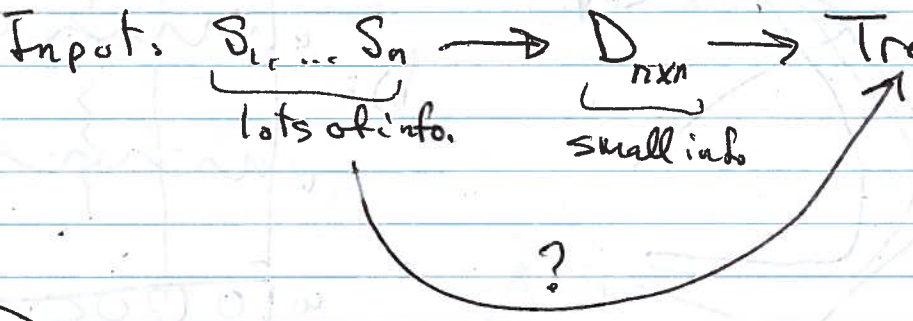
Problems with N-J algo

① If D is not ultrametric (D is not estimated ~~with~~ perfectly)

$\Rightarrow$ N-J comes with guarantees about optimality

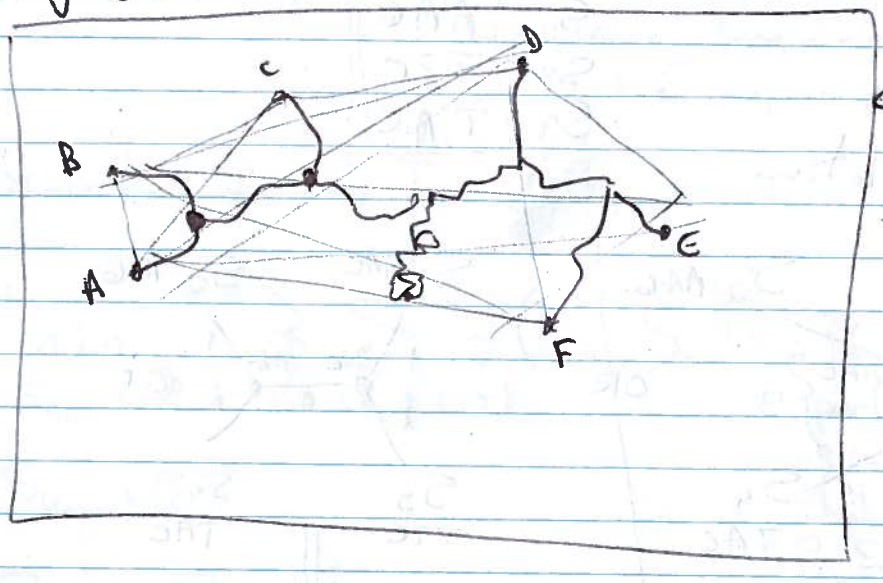
NO

② Information Bottleneck problem: Tree inferred may be inaccurate



③ Provide no info about ancestral sequence or about evolutionary events that led to $S_1 \ldots S_n$

Phylogenetic Inference - Maximum Parsimony



Large Parsimony Problem:

Given: $S_1 \ldots S_n$ of length L, in multiple sequence aln, with columns containing gaps removed

Find:
- Tree T with leaves labeled with $S_1 \ldots S_n$
- Ancestral sequence S_u for each internal node of T

such that $\sum_{(u,v) \in E(T)} d(S_u, S_v)$ is minimized

$d(S_u, S_v)$ = # of substitutions btw S_u, S_v

Example: Input:

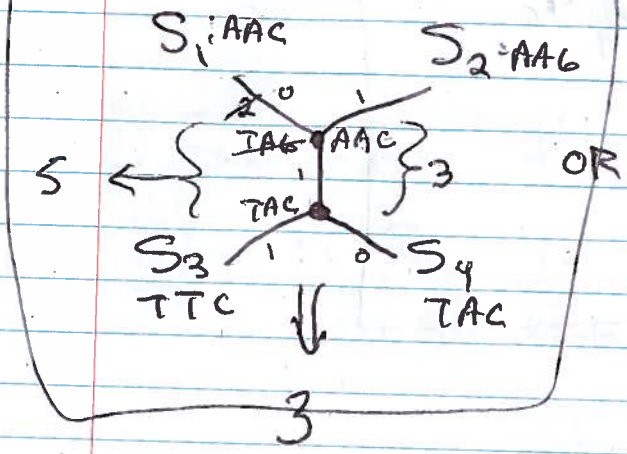
MSA

S_1: AAC
S_2: AA6
S_3: TTC
S_4: TAC

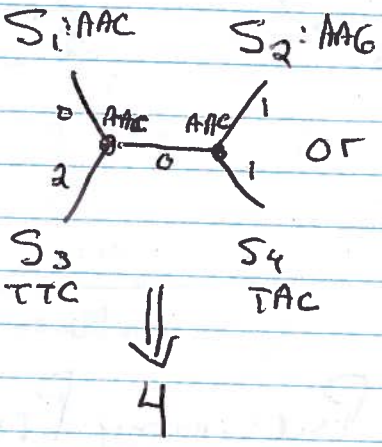
Unrooted tree

Rooted tree

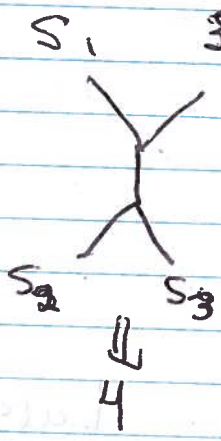
Opt-Solution



3

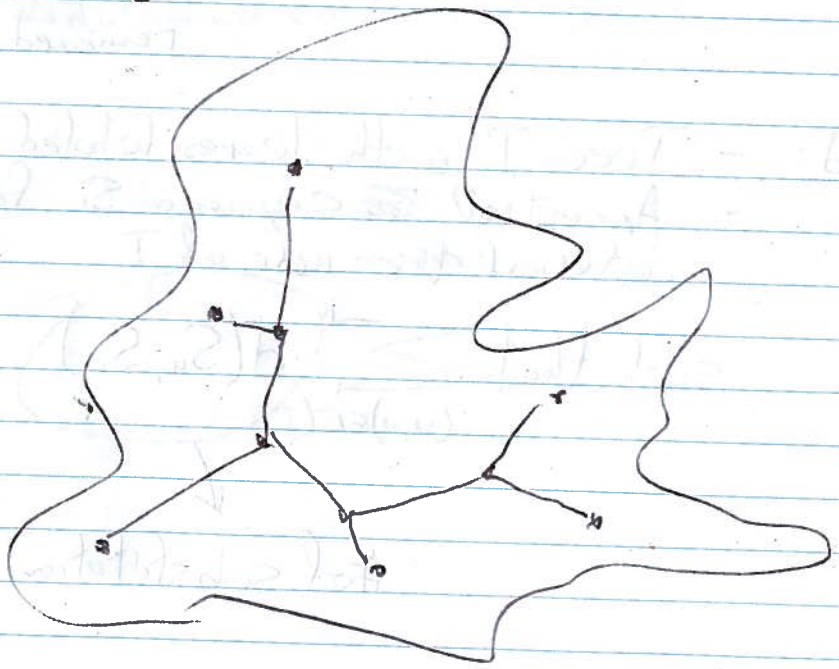


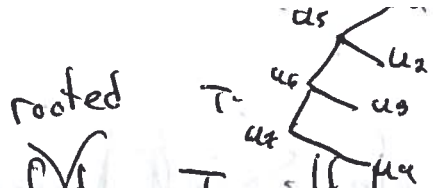
4



4

Steiner tree





Define Parsimony Score of tree T with leaves $u_1, u_2 \dots u_n$ and internal nodes $u_{n+1}, u_{n+2} \dots, u_{2n-1}$

$$\text{ParsScore}(S_1, S_2 \dots S_n, T) = \min_{\substack{S_{u_{n+1}} \in (\Sigma)^n \\ S_{u_{n+2}} \in (\Sigma)^n \\ \vdots \\ S_{u_{2n-1}} \in \Sigma^n}} \left\{ \sum_{(u,v) \in E(T)} d(S_u, S_v) \right\}$$

$\equiv$ The minimum # of substitutions performed along branches of T that can explain $S_1, S_2 \dots S_n$

$\left(4^L\right)^{n-1}$ cases $= 4^{L(n-1)}$

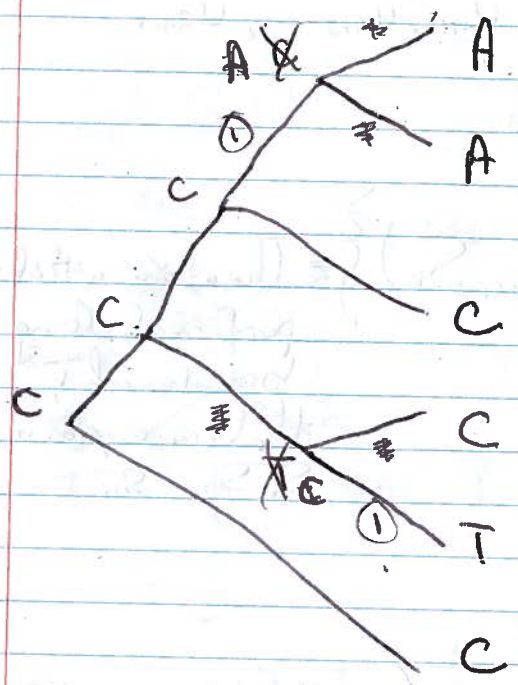
Observation: $\text{ParsScore}(S_1 \dots S_n, T) = \sum_{i=1}^{L} \text{ParsScore}(S_1[i], \dots, S_n[i], T)$

→ Small Parsimony Problem

Given: - Aligned $S_1, \dots, S_n$ of length L
- Phylogenetic tree T

Goal: Calculate $\text{ParsScore}(S_1 \dots S_n, T)$

Example: Small Parsimony Problem

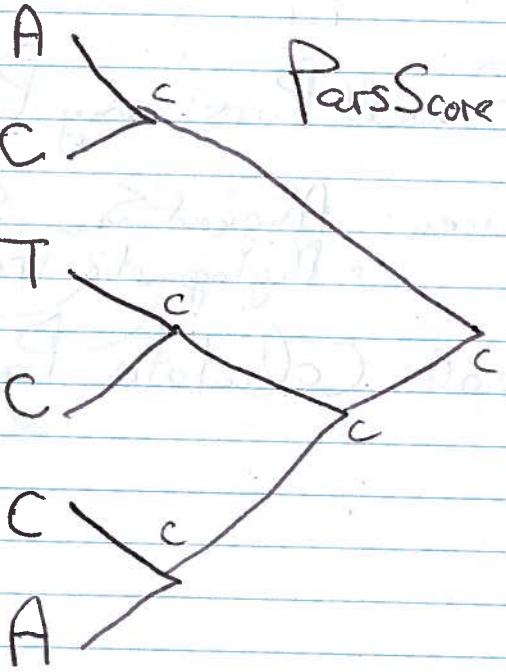


Score = 4

Score = 2 Optimal sol.

$\Rightarrow$ ParsScore(A, A, C, C, C, T, C, Tree) = 2

How to calculate $ParsScore[S_1[i], S_2[i], \ldots S_n[i], T]$?



ParsScore = 3

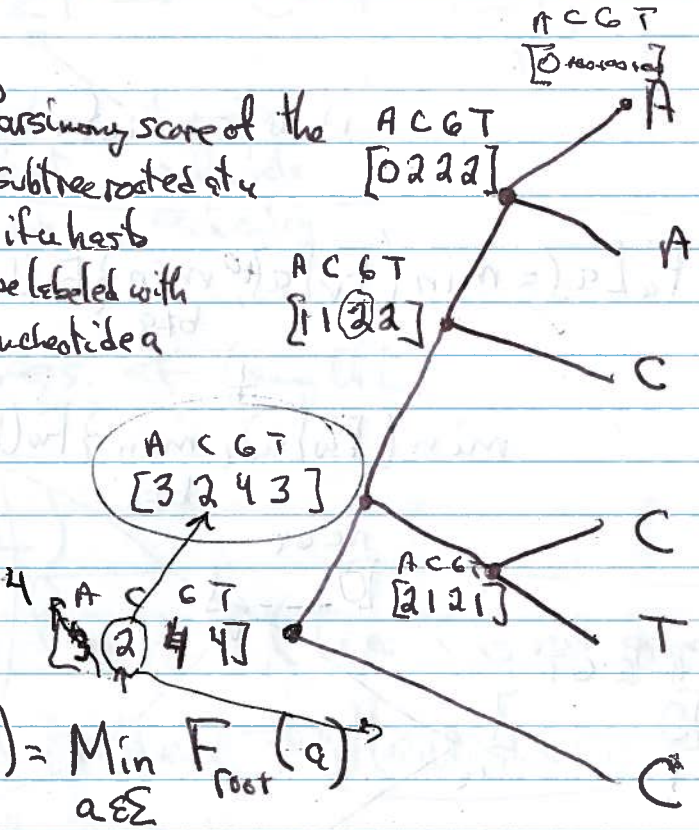
Sankoff Algorithm (1975)

Input: $\{S_1, S_2 \ldots S_n$ each of length 1
Tree T

Goal: Calculate ParsScore$(S_1 \ldots S_n, T)$

Define $F_u[a]$ = Parsimony score of the subtree rooted at u if u has b be labeled with nucleotide a

(u = node, $a \in \{A, C, G, T\}$)



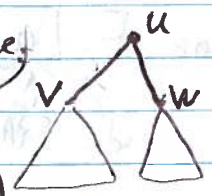
$$\text{ParsScore}(\text{Tree}) = \min_{a \in \Sigma} F_{\text{root}}(a)$$

Algo: For each node u in tree (from leaves back to the root)

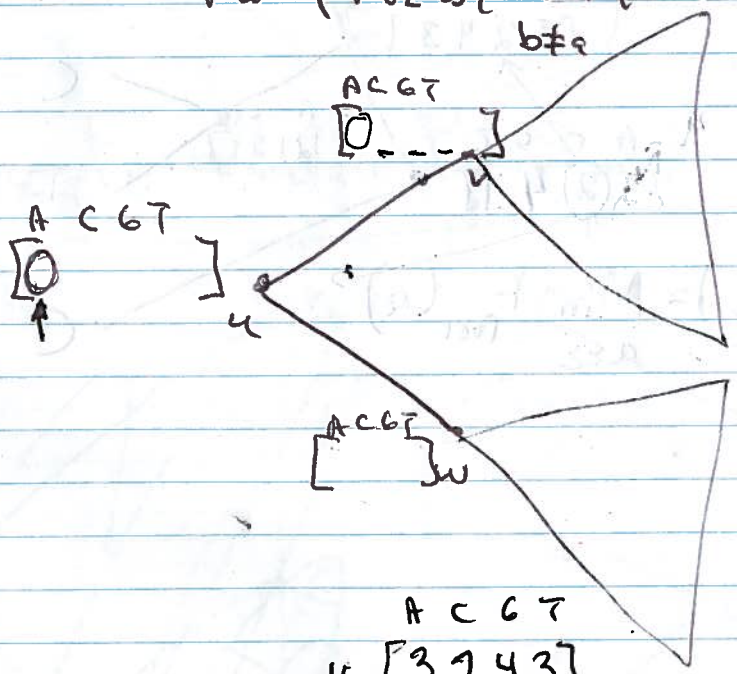
For each $a \in \{A, C, G, T\}$

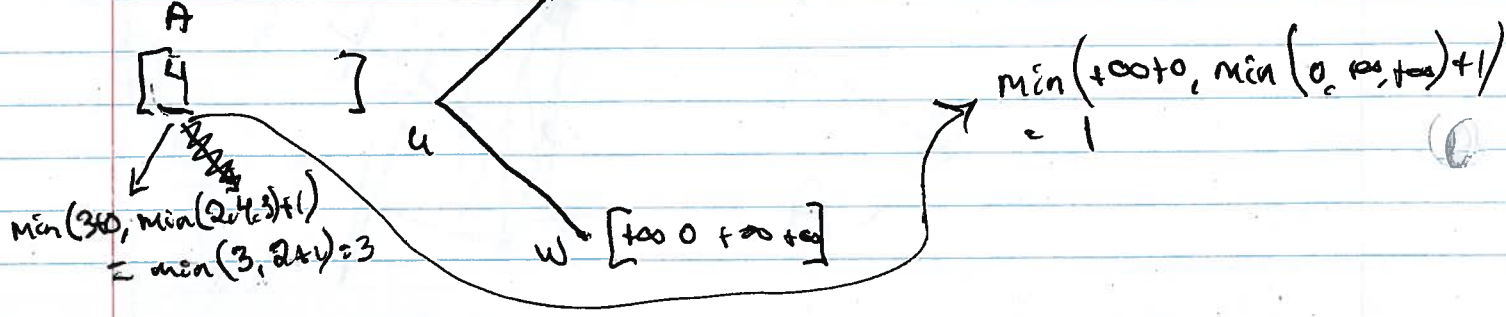
if u is a leaf $F_u[a] = \begin{cases} 0 \text{ if } s_u = a \\ +\infty \text{ if } s_u \neq a \end{cases}$

if v is an internal node,



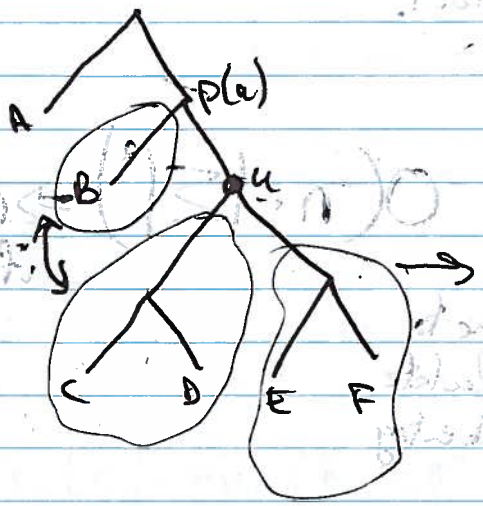
$$F_u[a] = \min\left(F_v[a], \min_{b \neq a}\{F_v(b)\} + 1\right) + \min\left(F_w[a], \min_{b \neq a}\{F_w(b)\} + 1\right)$$





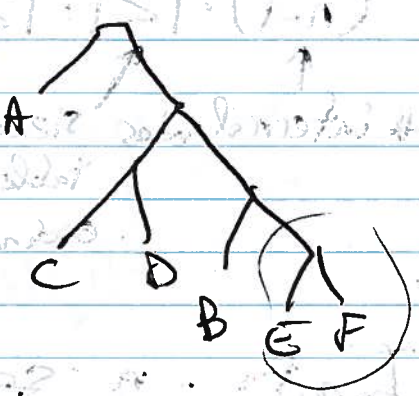
Tree neighborhood definitions

T:



Nearest-neighbor interchange (NNI)

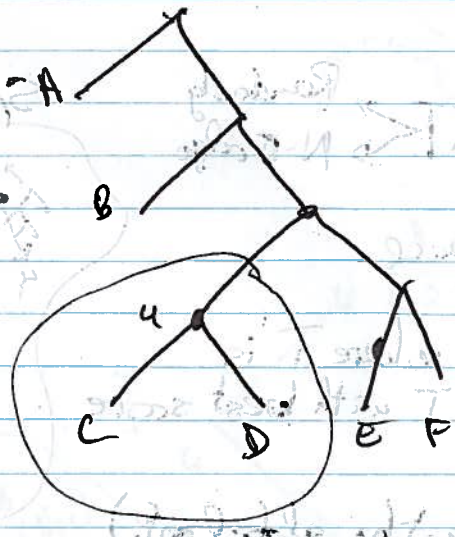
Pick some internal node u



$$Neighbors(T) = \{ T' \text{ s.t. } T' \text{ can be obtained from } T \text{ with } \underline{\text{one}} \text{ NNI} \}$$

$$|Neighbors(T)| = (n-2) \cdot 2$$

Subtree Prune and Regraft



Choose any node in T
- cut in off from T
- reinsert along other edge of T

⇒