CS 466
Introduction to Bioinformatics
Lecture 15

Mohammed El-Kebir

October 14, 2020

Course Announcements

HW 3 will be released Oct 23 – due Oct 31 by 11:59pm

Project proposal due on Nov 5 by 11:59pm

(Motivation, Datasets/papers, Planned method/experiments, Timeline)

Project report due on Dec 20

Outline

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Hierarchical clustering
- Additive distance phylogeny
- Four point condition
- Neighbor joining

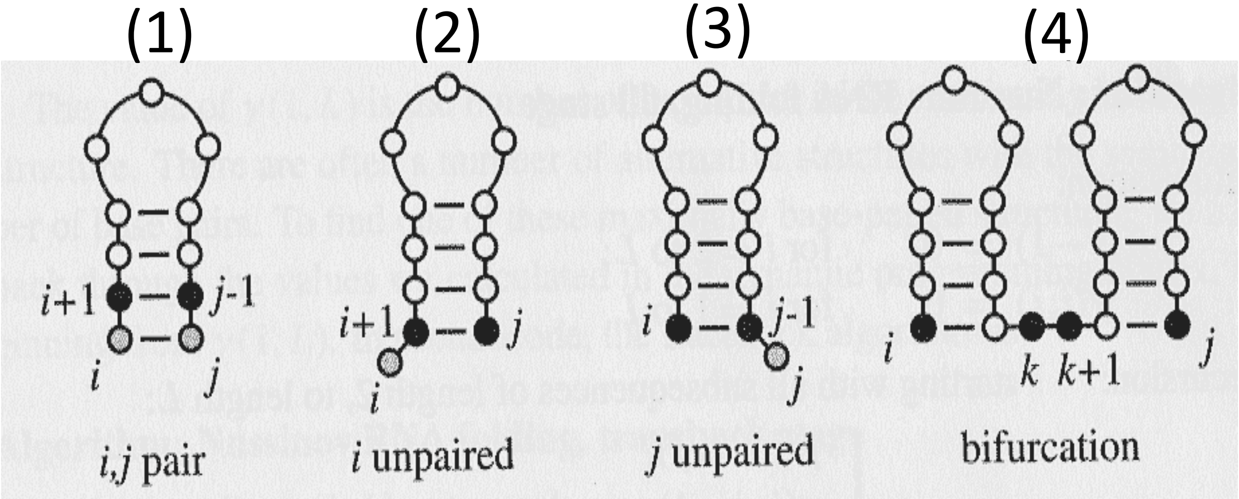
Reading:

- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

Nussinov Algorithm – Dynamic Programming

Problem: Given RNA sequence $\mathbf{v} \in \{\mathrm{A}, \mathrm{U}, \mathrm{C}, \mathrm{G}\}^n$, find a *pseudoknot-free secondary structure* with the maximum number of complementary base pairings

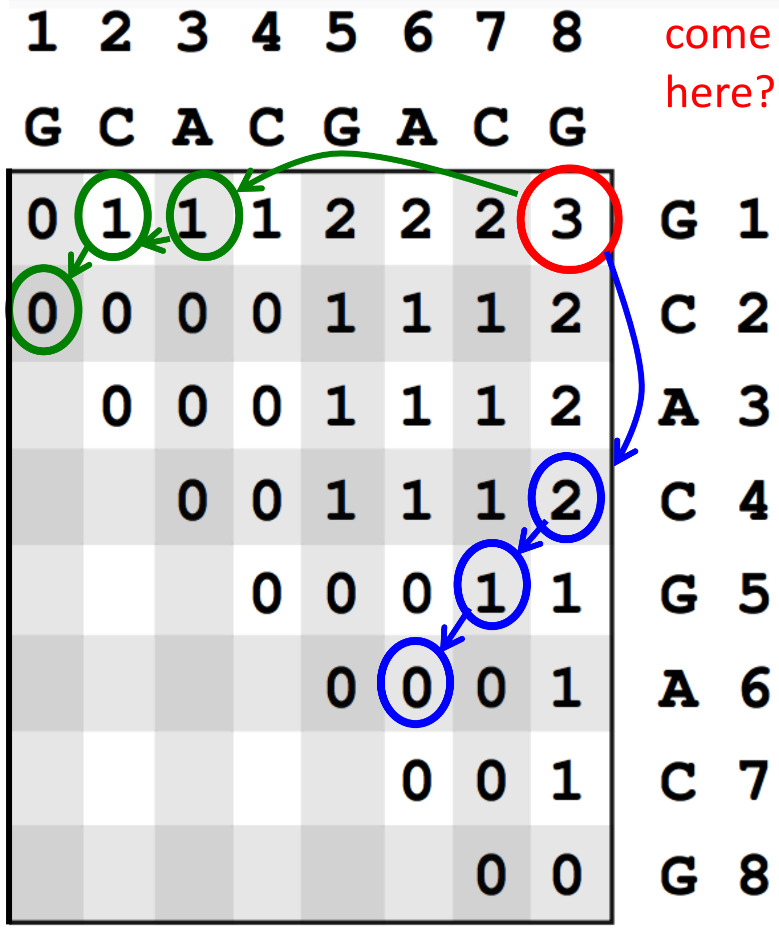
Let $s[i, j]$ denote the maximum number of pseudoknot-free complementary base pairings in subsequence $v_i, \dots, v_j$



$$s[i,j] = \max \begin{cases} 0, & \text{if } i \geq j, \\ s[i+1, j-1] + 1, & \text{if } i < j \text{ and } (v_i, v_j) \in \Gamma, \quad \text{(1)} \\ s[i+1, j-1], & \text{if } i < j \text{ and } (v_i, v_j) \notin \Gamma, \quad \text{(1*)} \\ s[i+1, j], & \text{if } i < j, \quad \text{(2)} \\ s[i, j-1], & \text{if } i < j, \quad \text{(3)} \\ \max_{i<k<j}\{s[i,k] + s[k+1, j]\}, & \text{if } i < j, \quad \text{(4)} \end{cases}$$

Question: Which case is redundant?

Nussinov Algorithm – Example With Bifurcation

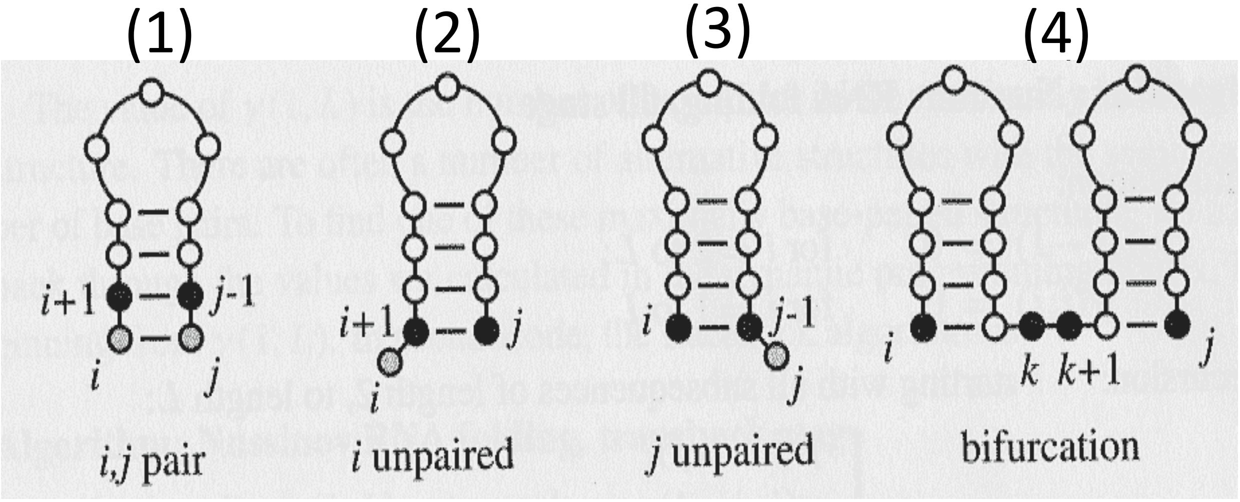


Where did we come from to get here?

	1	2	3	4	5	6	7	8		
	G	C	A	C	G	A	C	G		
	0	1	1	1	2	2	2	3	G	1
	0	0	0	0	1	1	1	2	C	2
		0	0	0	1	1	1	2	A	3
			0	0	1	1	1	2	C	4
				0	0	0	1	1	G	5
					0	0	0	1	A	6
						0	0	1	C	7
							0	0	G	8

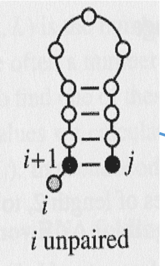
GCACGACG

( ) . ((.))

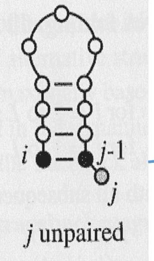


$$
s[i,j] = \max \begin{cases} 0, & \text{if } i \geq j, \\ s[i+1, j-1] + 1, & \text{if } i < j \text{ and } (v_i, v_j) \in \Gamma, \quad (1) \\ s[i+1, j-1], & \text{if } i < j \text{ and } (v_i, v_j) \notin \Gamma, \quad (1^*) \\ s[i+1, j], & \text{if } i < j, \quad (2) \\ s[i, j-1], & \text{if } i < j, \quad (3) \\ \max_{i<k<j} \{ s[i,k] + s[k+1, j] \}, & \text{if } i < j, \quad (4) \end{cases}
$$

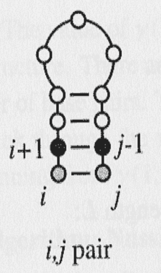
Nussinov Algorithm – Traceback Step



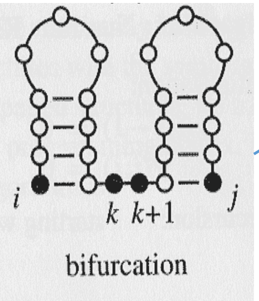
i unpaired



j unpaired



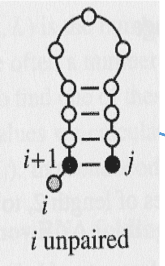
i,j pair

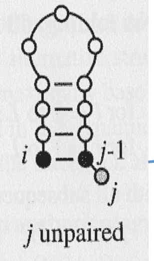


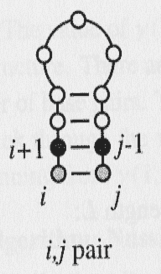
bifurcation

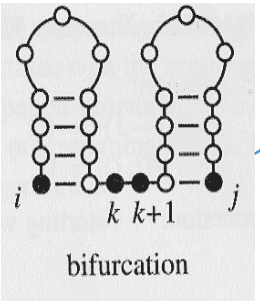
```
Push (1, n) onto stack
Repeat until stack is empty:
        pop (i,j)
        if i ≥ j continue
        else if s[i+1,j] = s[i,j]
                push (i+1,j)
        else if s[i,j-1] = S[i,j]
                push (i,j-1)
        else if s[i+1,j-1] + 1 = s[i,j]
                record (i,j) base pair
                push (i+1,j-1)
        else for k = i+1 to j-1
                if s[i,k]+s[k+1,j] = s[i,j]
                        push (k+1,j)
                        push (i,k)
                    break (for loop)
```

Nussinov Algorithm – Traceback Step









```
Push (1, n) onto stack
Repeat until stack is empty:
        pop (i,j)
        if i ≥ j continue
        else if s[i+1,j] = s[i,j]
                push (i+1,j)
        else if s[i,j-1] = S[i,j]
                push (i,j-1)
        else if s[i+1,j-1] + 1 = s[i,j]
                record (i,j) base pair
                push (i+1,j-1)
        else for k = i+1 to j-1
                if s[i,k]+s[k+1,j] = s[i,j]
                        push (k+1,j)
                        push (i,k)
                    break (for loop)
```

```
BackTrack(i, j)
if i < j
        if s[i+1, j] = s[i, j]
                BackTrack(i+1, j)
        else if s[i, j-1] = S[i, j]
                BackTrack(i, j-1)
        else if s[i+1,j-1] + 1 = s[i, j]
                Output (i, j)
                BackTrack(i+1, j-1)
        else for k = i+1 to j-1
                if s[i, k]+s[k+1, j] = s[i, j]
                        BackTrack(k+1, j)
                        BackTrack(i, k)
                    break (for loop)
```

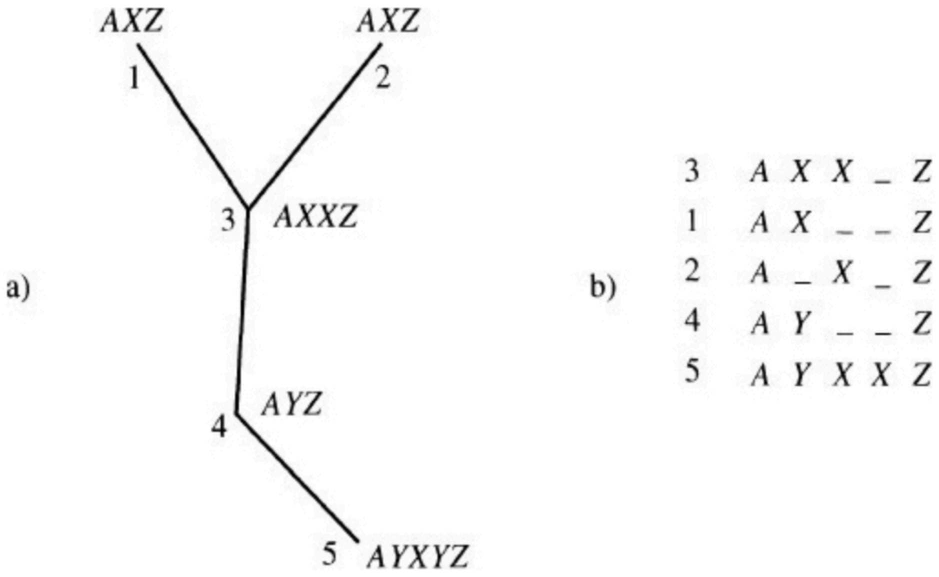
Outline

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Hierarchical clustering
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

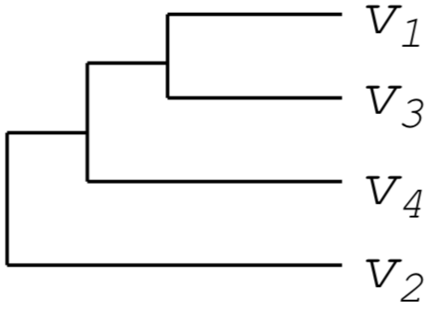
- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

Alignments and Trees



```
3  A X X _ Z
1  A X _ _ Z
2  A _ X _ Z
4  A Y _ _ Z
5  A Y X X Z
```

Tree / star alignment



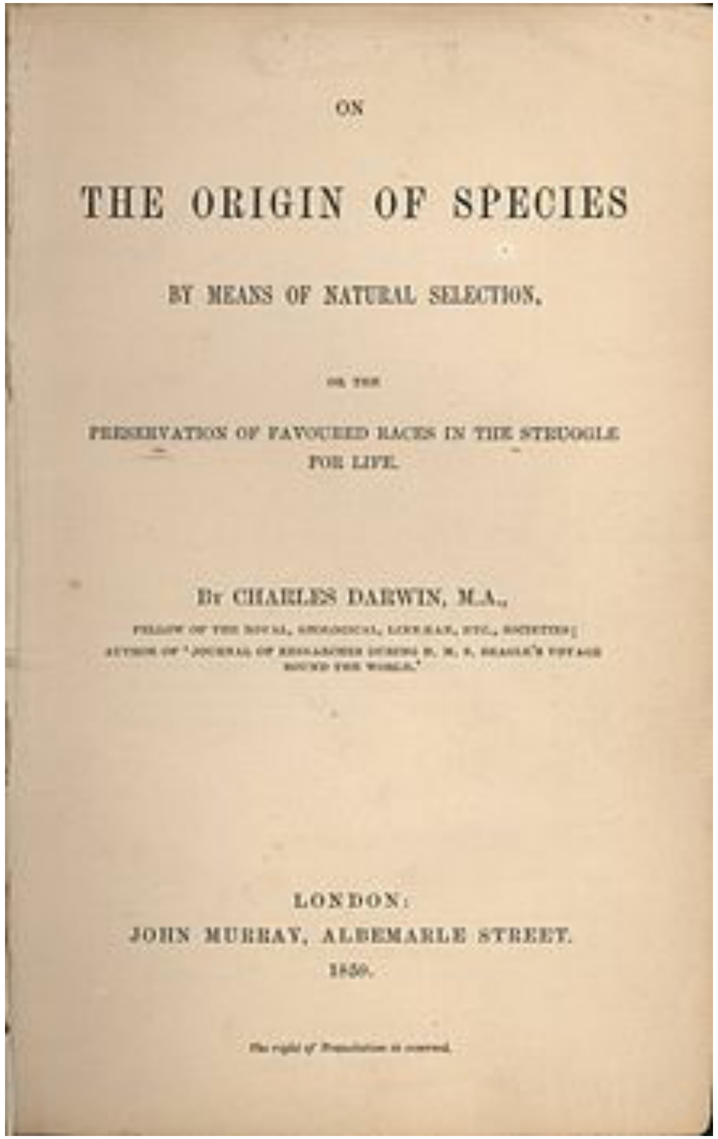
Guide tree in
progressive alignment

Tree topology represents similarity/distance between sequences

Biological sequences typically come from the present

Evolutionary Studies and Phylogenies

- Since Darwin's book (1859) until 1960s: Phylogeny reconstruction from anatomical features
- Subjective observations led to inconclusive/incorrect phylogenies



ON

THE ORIGIN OF SPECIES

BY MEANS OF NATURAL SELECTION,

OR THE

PRESERVATION OF FAVOURED RACES IN THE STRUGGLE
FOR LIFE.

BY CHARLES DARWIN, M.A.,

FELLOW OF THE ROYAL, GEOLOGICAL, LINNÆAN, ETC., SOCIETIES;
AUTHOR OF 'JOURNAL OF RESEARCHES DURING H. M. S. BEAGLE'S VOYAGE
ROUND THE WORLD.'

LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.

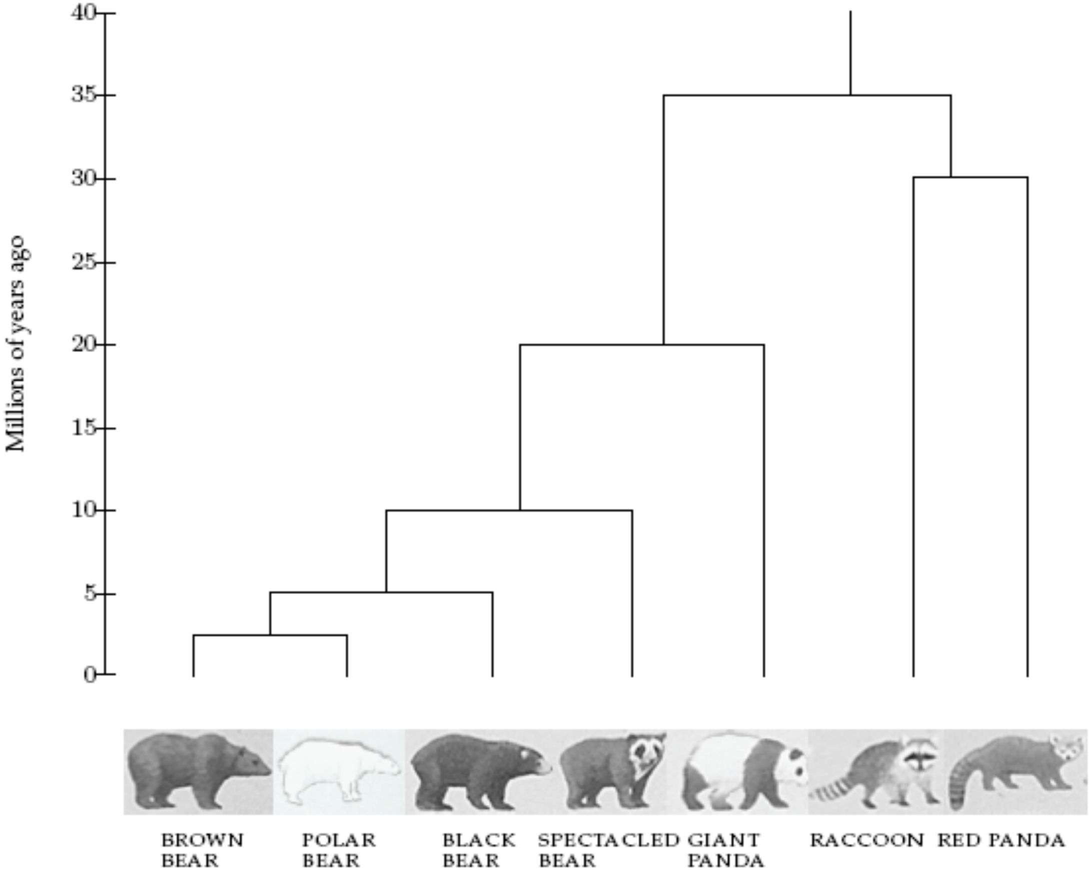
The right of Translation is reserved.

Evolutionary Studies and Phylogenies

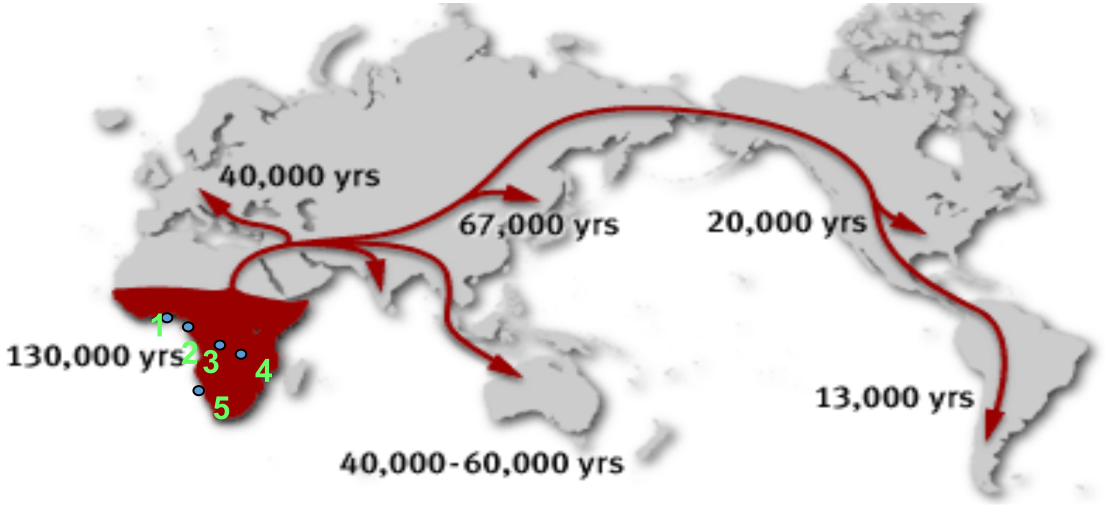
- Subjective observations led to inconclusive/incorrect phylogenies

<u>Example</u>

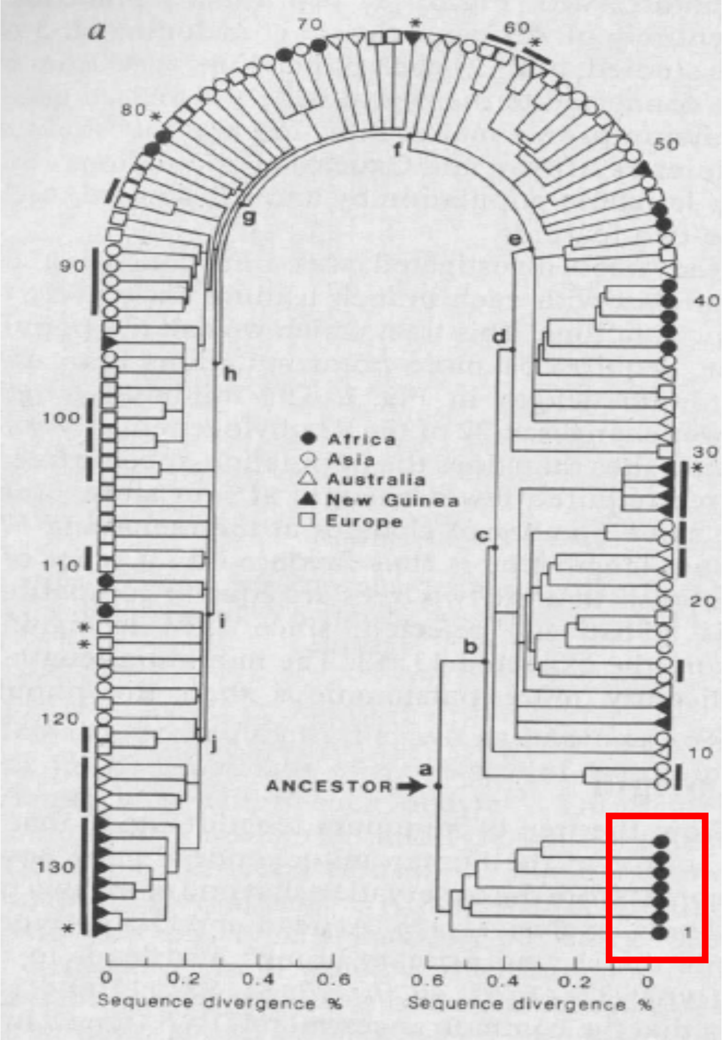
- Giant pandas look like bears but have features that are unusual for bears and typical for racoons
- In 1985, Steven O'Brien and colleagues solved the giant panda classification problem using DNA sequences and algorithms



Out of Africa Hypothesis



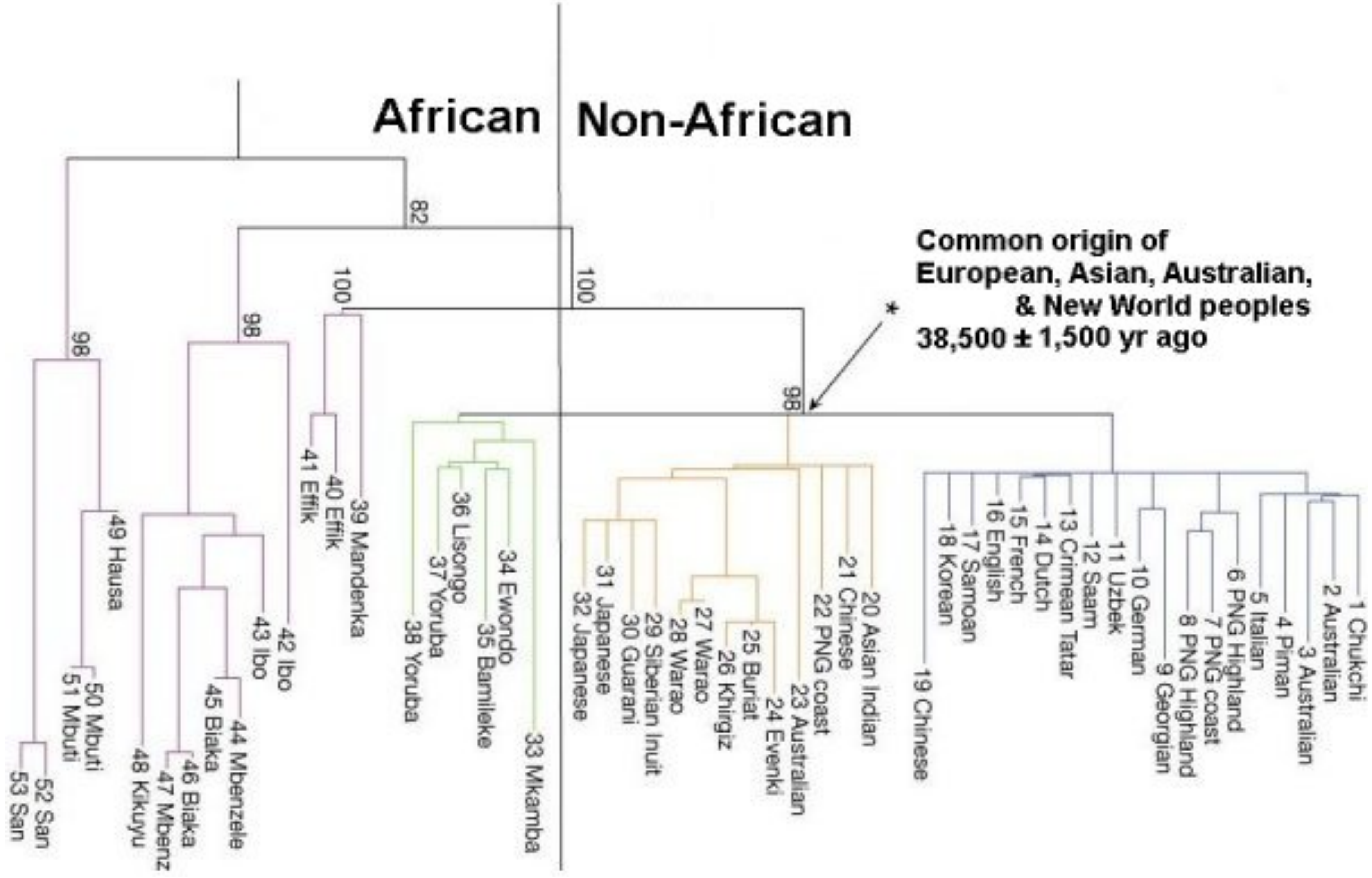
http://www.becominghuman.org



Vigilant, Stoneking, Harpending, Hawkes, and Wilson (1991)

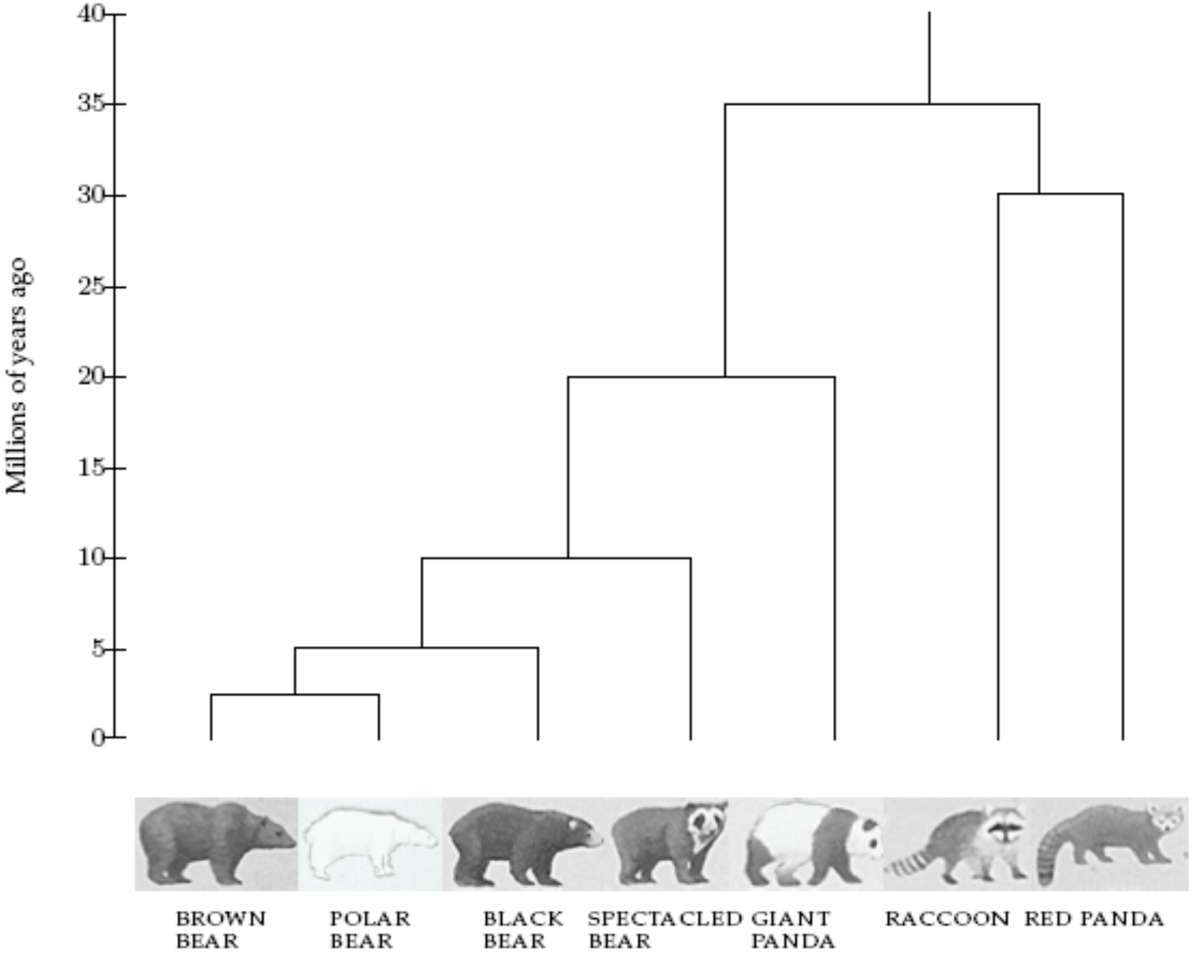
Out of Africa Hypothesis claims that our most ancient ancestor lived in Africa roughly 200,000 years ago

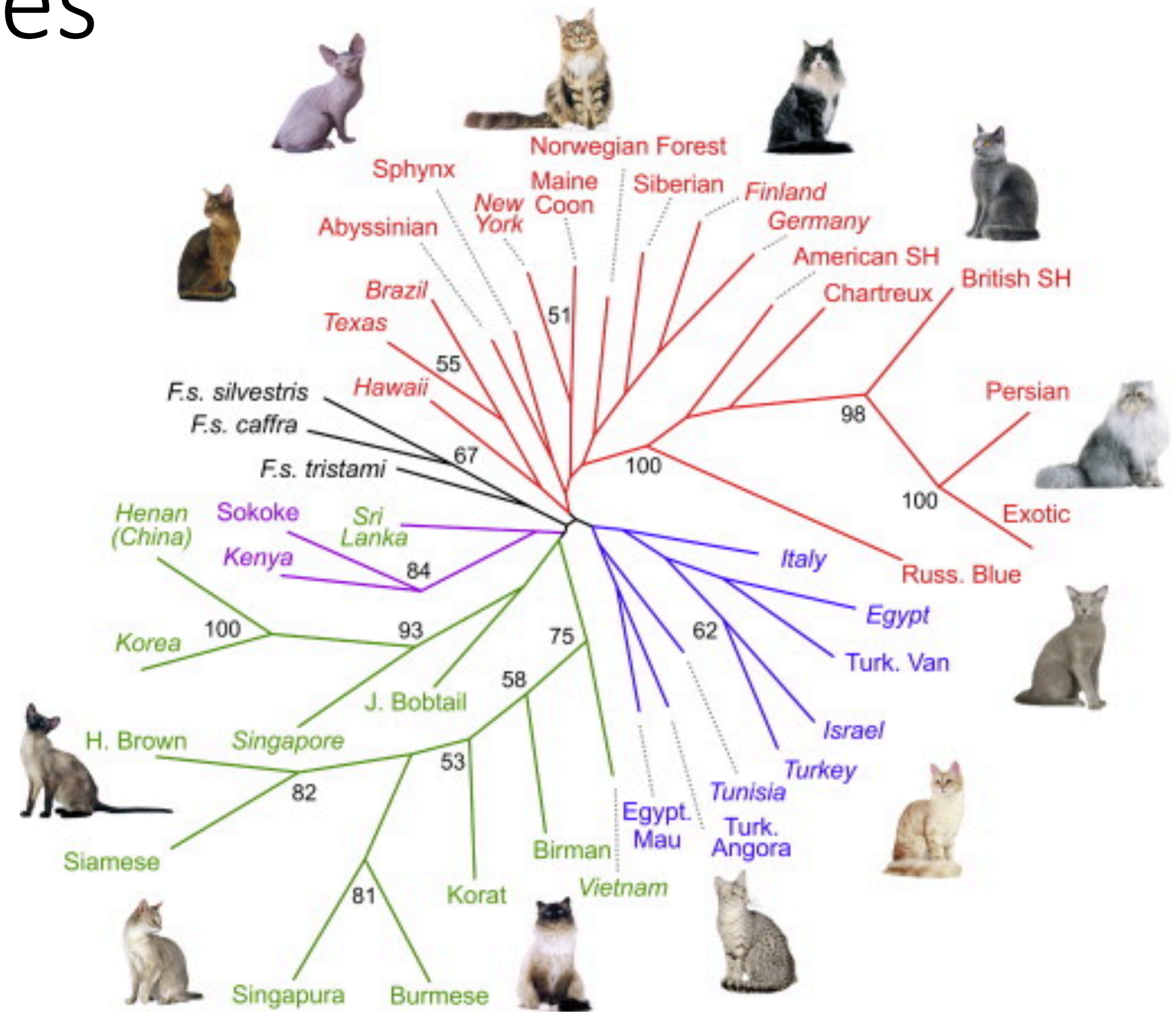
Evolutionary Tree of Humans



Common origin of
European, Asian, Australian,
& New World peoples
38,500 ± 1,500 yr ago

Evolutionary Tree of Species



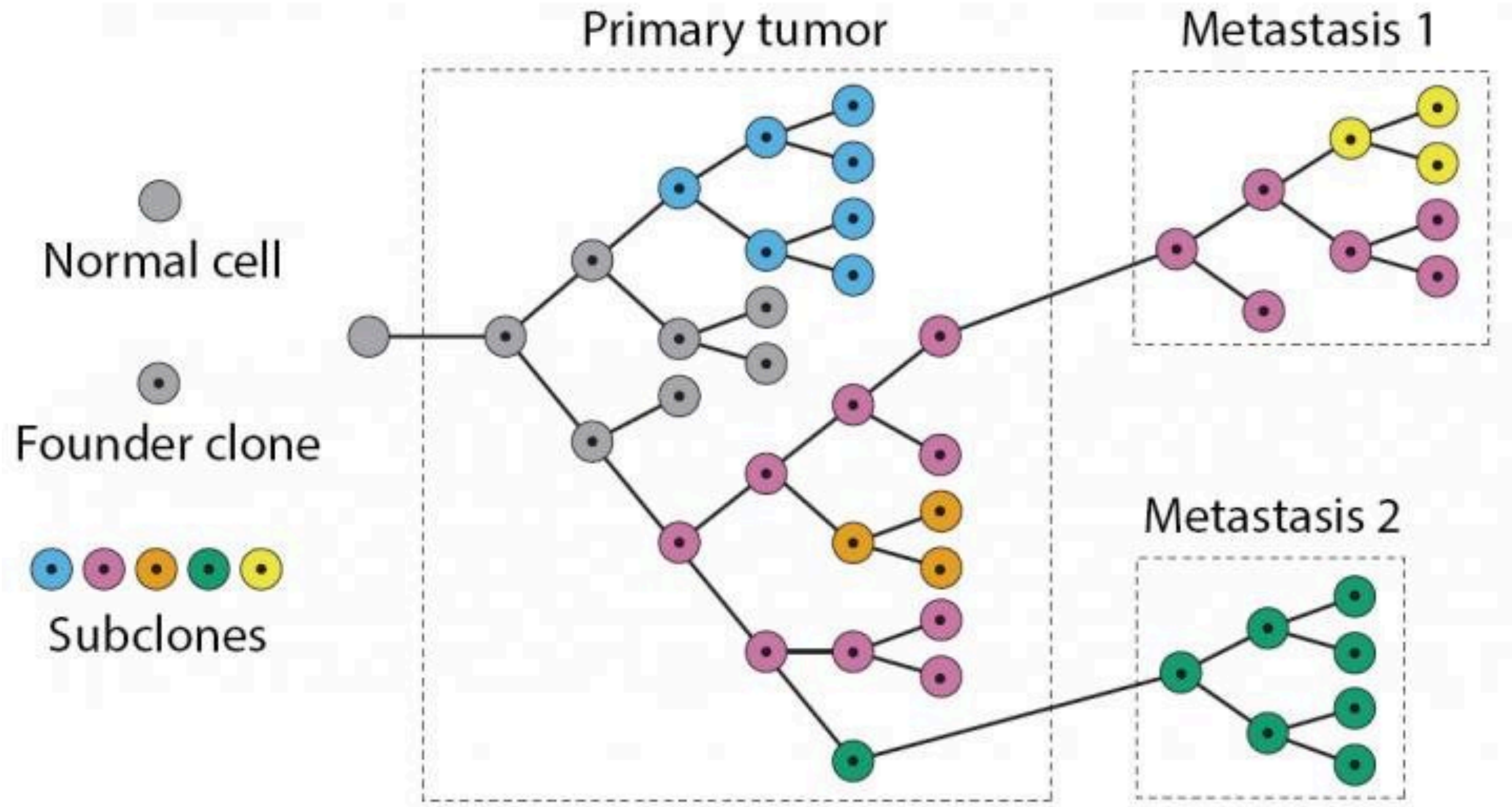


http://bix.ucsd.edu/bioalgorithms/

[Lipinski *et al.*, 2008]

Question: What are the evolutionary relationships between species?

Evolutionary Tree of a Tumor



https://www.sciencedaily.com/releases/2016/09/160909223504.htm

Phylogenetic Tree Reconstruction

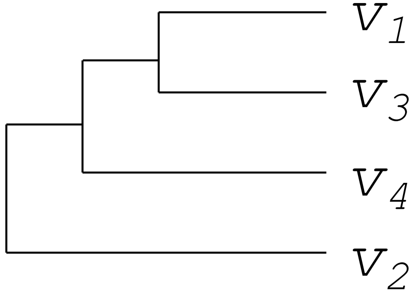
```
Mouse:      ACAGTGACGCCACACACGT
Gorilla:    CCTGTGACGTAACAAACGA
Chimpanzee: CCTGTGAGGTAGCAAACGA
Human:      CCTGTGAGGTAGCACACGA
```

Distance Metric

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–

Distance Table

???



Phylogenetic Tree

Question: Given sequence data, how to reconstruct tree?

Outline

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Hierarchical clustering
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

Clustering

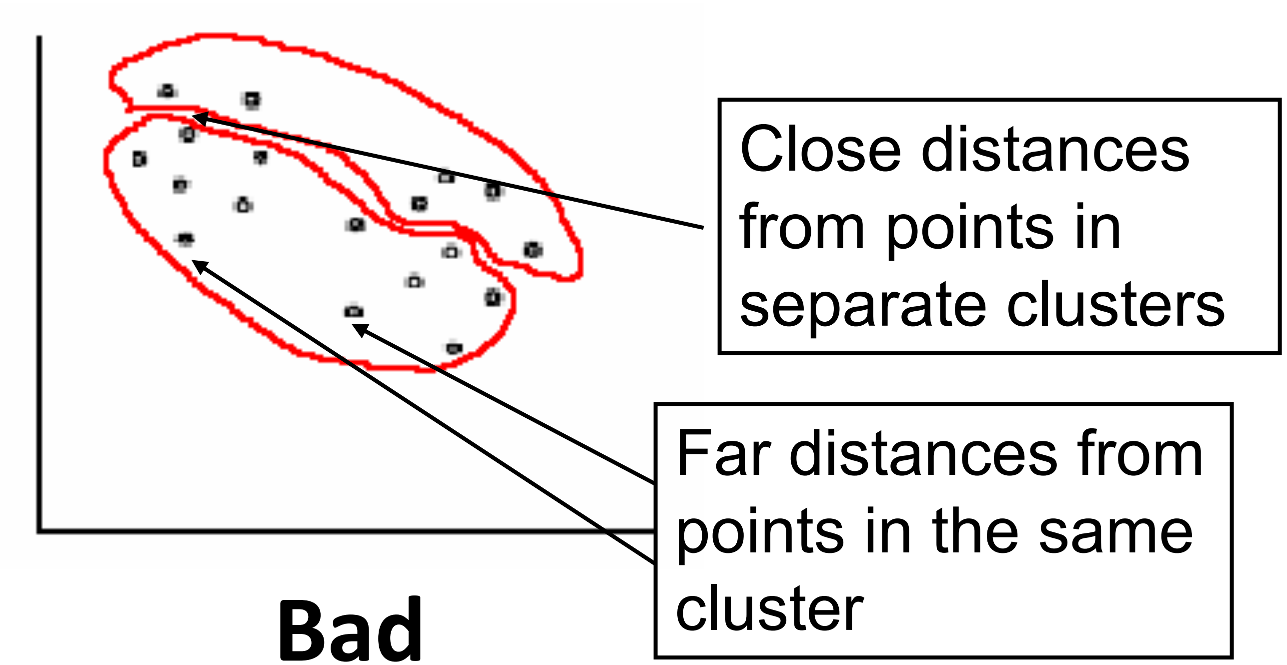
Given:

(1) $n \times n$ matrix $D = [d_{i,j}]$

Want:

(1) Homogeneity within clusters

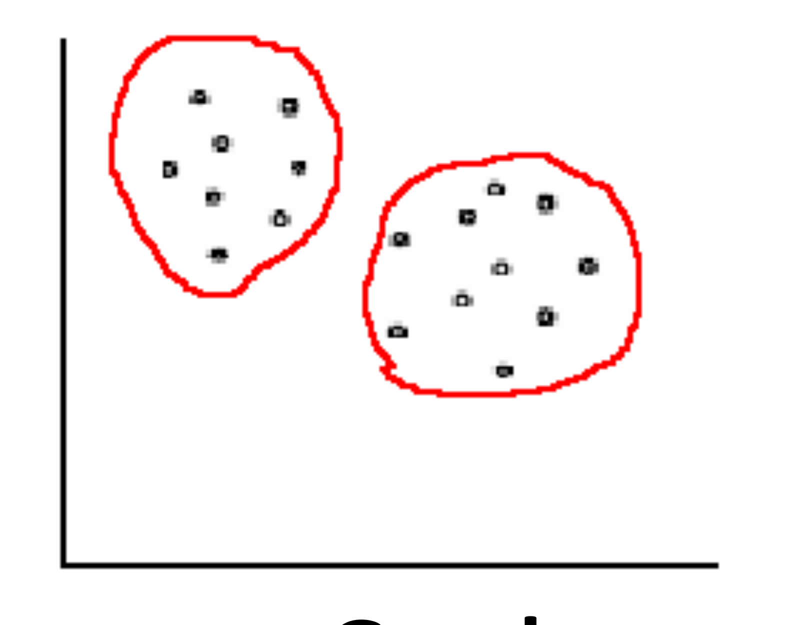
(2) Separation between clusters



Close distances from points in separate clusters

Far distances from points in the same cluster

Bad

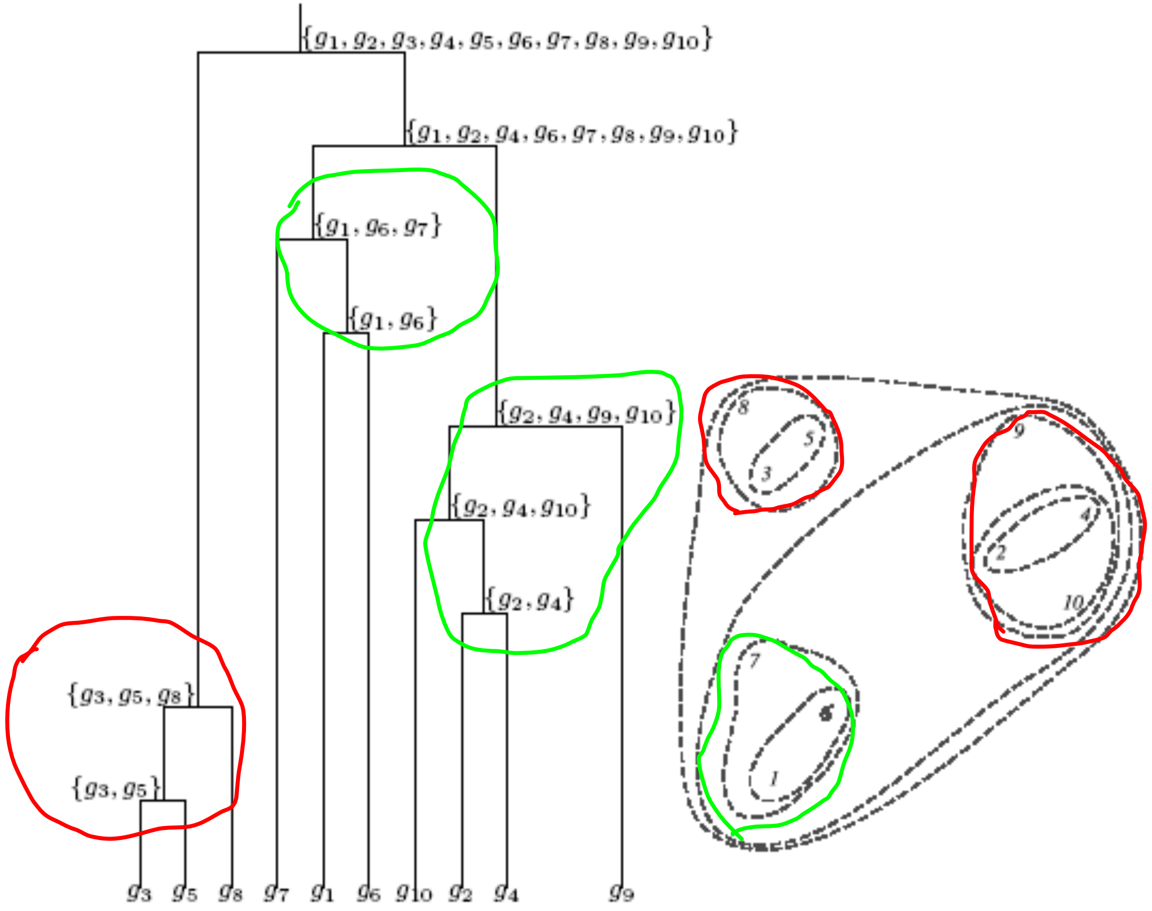


Good

Hierarchical Clustering

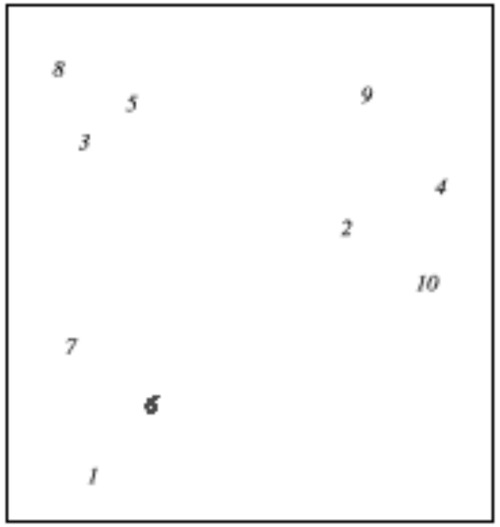
Organize elements into a tree such that:

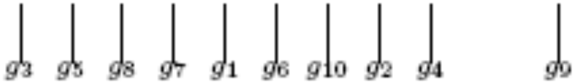
- Leaves are elements
- Paths between leaves represent pairwise element distance
- Similar elements lie within same subtrees


$\{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_2, g_4, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_6, g_7\}$
$\{g_1, g_6\}$
$\{g_2, g_4, g_9, g_{10}\}$
$\{g_2, g_4, g_{10}\}$
$\{g_2, g_4\}$
$\{g_3, g_5, g_8\}$
$\{g_3, g_5\}$
g_3 g_5 g_8 g_7 g_1 g_6 g_{10} g_2 g_4 g_9
8
5
3
9
4
2
10
7
6
1

Hierarchical Clustering

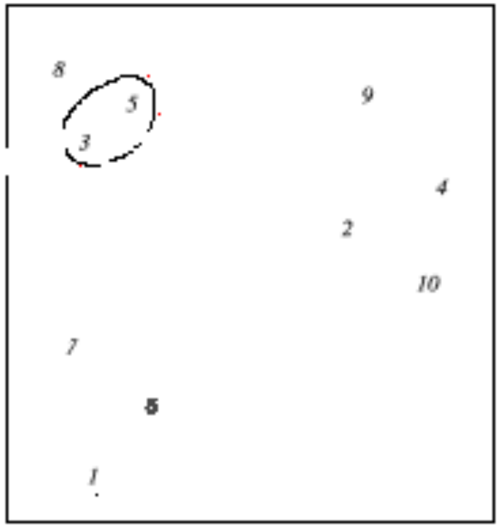
```
Hierarchical Clustering (D , n)
    Form n clusters each with one element
    Construct a graph T by assigning one vertex to each cluster
    while there is more than one cluster
        Find the two closest clusters $C_1$ and $C_2$
        Merge $C_1$ and $C_2$ into new cluster C with $|C_1| + |C_2|$ elements
        Compute distance from C to all other clusters
        Add a new vertex C to T and connect to vertices $C_1$ and $C_2$
        Remove rows and columns of D corresponding to $C_1$ and $C_2$
        Add a row and column to D corresponding to the new cluster C
    return T
```

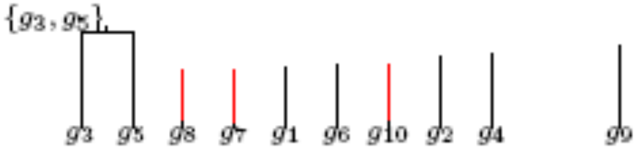




Hierarchical Clustering

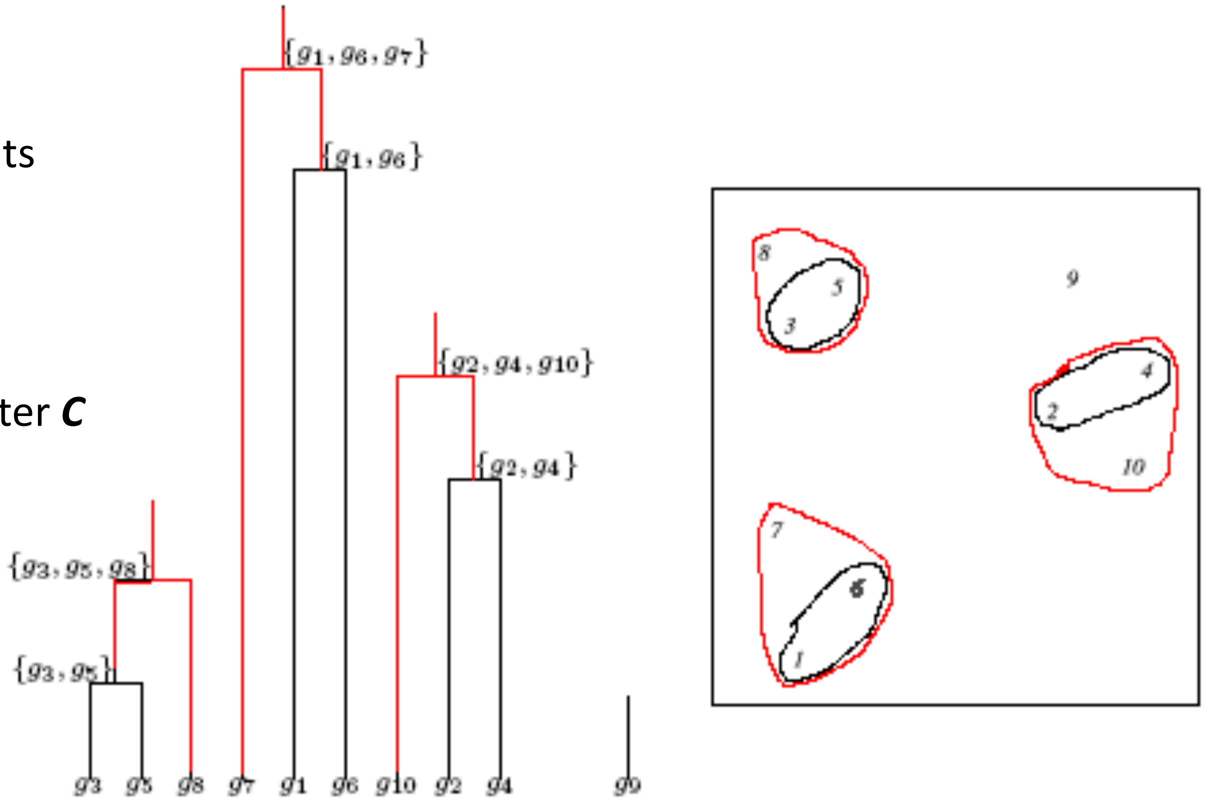
```
Hierarchical Clustering ($D$ , $n$)
  Form $n$ clusters each with one element
  Construct a graph $T$ by assigning one vertex to each cluster
  while there is more than one cluster
    Find the two closest clusters $C_1$ and $C_2$
    Merge $C_1$ and $C_2$ into new cluster $C$ with $|C_1| + |C_2|$ elements
    Compute distance from $C$ to all other clusters
    Add a new vertex $C$ to $T$ and connect to vertices $C_1$ and $C_2$
    Remove rows and columns of $D$ corresponding to $C_1$ and $C_2$
    Add a row and column to $D$ corresponding to the new cluster $C$
  return $T$
```





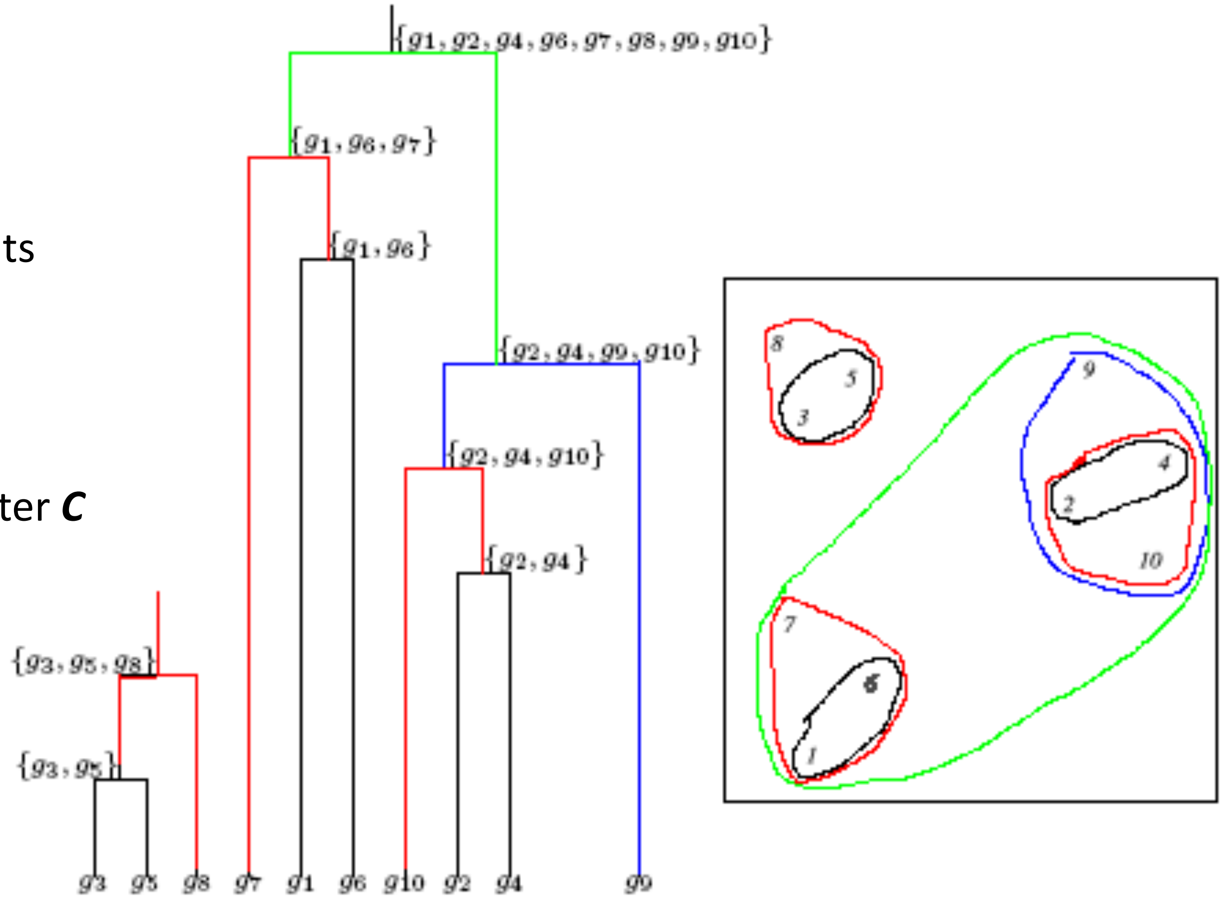
Hierarchical Clustering

```
Hierarchical Clustering ($D$ , $n$)
  Form $n$ clusters each with one element
  Construct a graph $T$ by assigning one vertex to each cluster
  while there is more than one cluster
    Find the two closest clusters $C_1$ and $C_2$
    Merge $C_1$ and $C_2$ into new cluster $C$ with $|C_1| + |C_2|$ elements
    Compute distance from $C$ to all other clusters
    Add a new vertex $C$ to $T$ and connect to vertices $C_1$ and $C_2$
    Remove rows and columns of $D$ corresponding to $C_1$ and $C_2$
    Add a row and column to $D$ corresponding to the new cluster $C$
  return $T$
```



Hierarchical Clustering

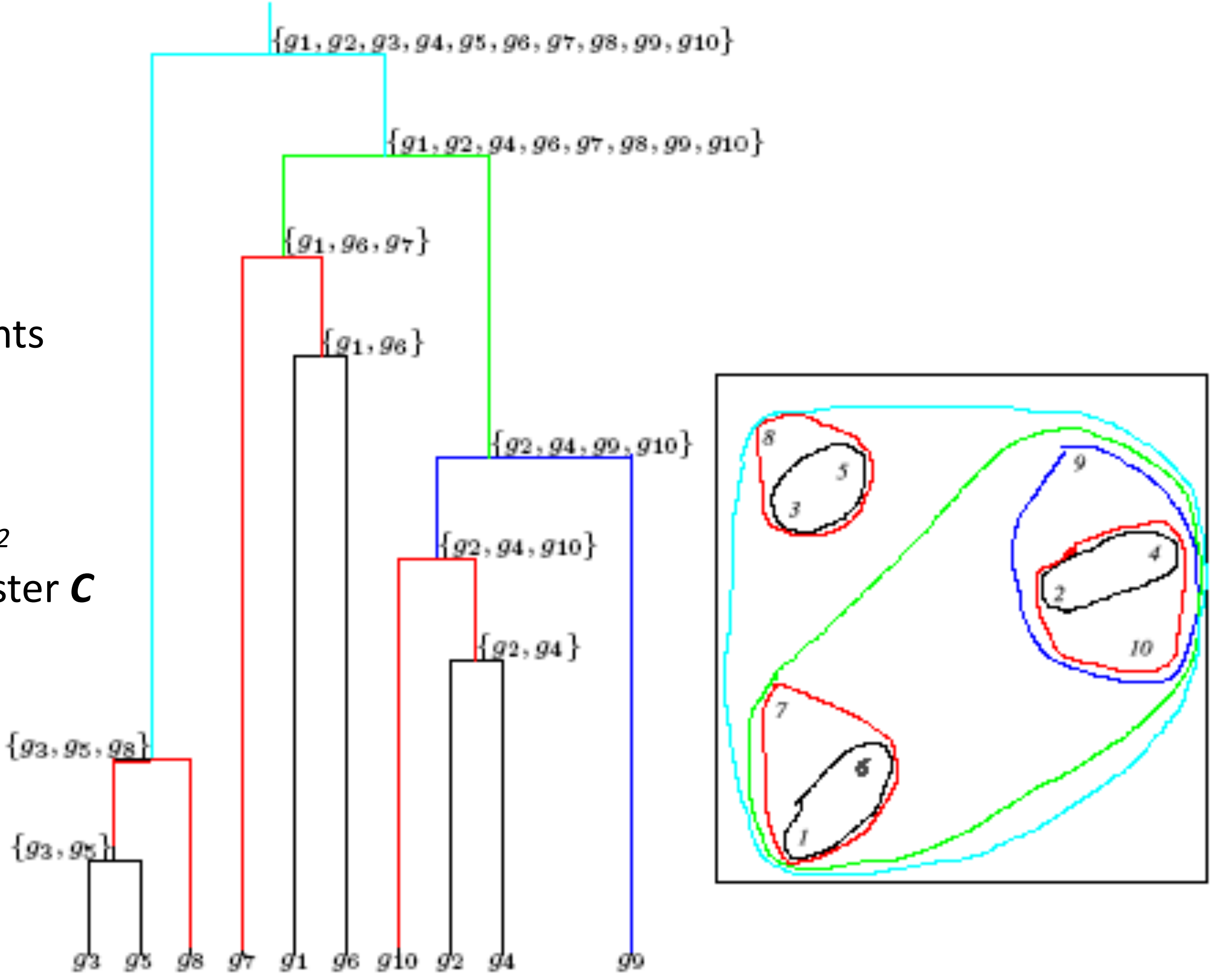
```
Hierarchical Clustering ($D$ , $n$)
    Form $n$ clusters each with one element
    Construct a graph $T$ by assigning one vertex to each cluster
    while there is more than one cluster
        Find the two closest clusters $C_1$ and $C_2$
        Merge $C_1$ and $C_2$ into new cluster $C$ with $|C_1| + |C_2|$ elements
        Compute distance from $C$ to all other clusters
        Add a new vertex $C$ to $T$ and connect to vertices $C_1$ and $C_2$
        Remove rows and columns of $D$ corresponding to $C_1$ and $C_2$
        Add a row and column to $D$ corresponding to the new cluster $C$
    return $T$
```



$\{g_1, g_2, g_4, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_6, g_7\}$
$\{g_1, g_6\}$
$\{g_2, g_4, g_9, g_{10}\}$
$\{g_2, g_4, g_{10}\}$
$\{g_2, g_4\}$
$\{g_3, g_5, g_8\}$
$\{g_3, g_5\}$
g_3 g_5 g_8 g_7 g_1 g_6 g_{10} g_2 g_4 g_9
8
5
3
9
4
2
10
7
6
1

Hierarchical Clustering

```
Hierarchical Clustering (D , n)
    Form n clusters each with one element
    Construct a graph T by assigning one vertex to each cluster
    while there is more than one cluster
        Find the two closest clusters $C_1$ and $C_2$
        Merge $C_1$ and $C_2$ into new cluster C with $|C_1| + |C_2|$ elements
        Compute distance from C to all other clusters
        Add a new vertex C to T and connect to vertices $C_1$ and $C_2$
        Remove rows and columns of D corresponding to $C_1$ and $C_2$
        Add a row and column to D corresponding to the new cluster C
    return T
```

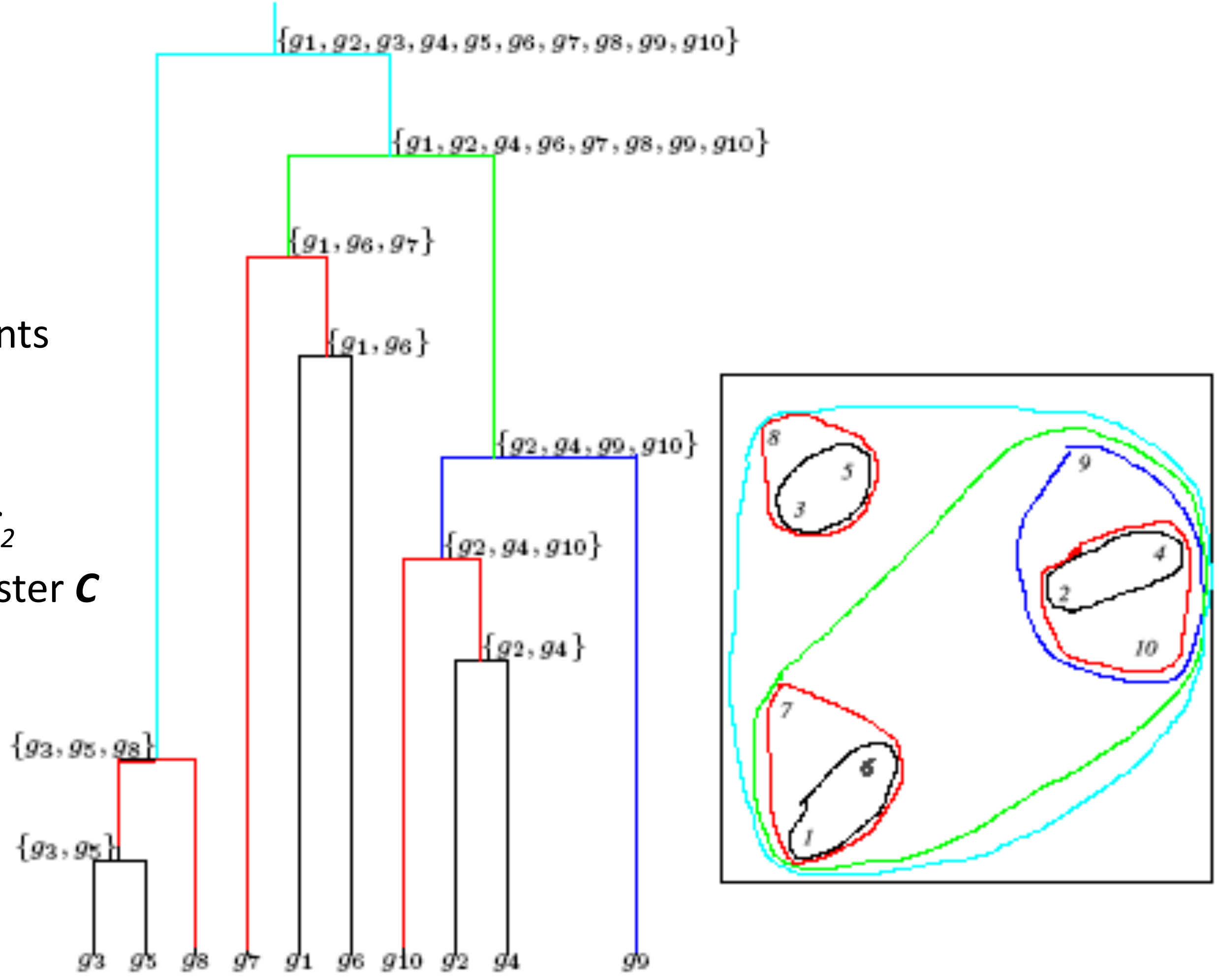


$\{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_2, g_4, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_6, g_7\}$
$\{g_1, g_6\}$
$\{g_2, g_4, g_9, g_{10}\}$
$\{g_2, g_4, g_{10}\}$
$\{g_2, g_4\}$
$\{g_3, g_5, g_8\}$
$\{g_3, g_5\}$
g_3 g_5 g_8 g_7 g_1 g_6 g_{10} g_2 g_4 g_9
8
5
3
9
4
2
10
7
6
1

Hierarchical Clustering

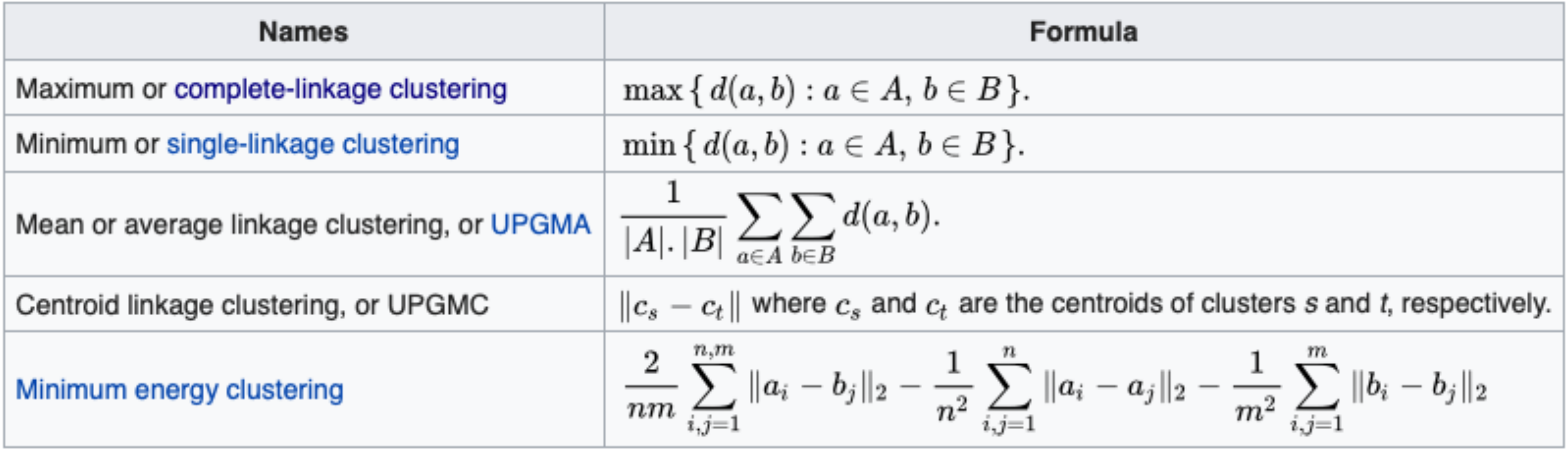
```
Hierarchical Clustering (D , n)
  Form n clusters each with one element
  Construct a graph T by assigning one vertex to each cluster
  while there is more than one cluster
    Find the two closest clusters $C_1$ and $C_2$
    Merge $C_1$ and $C_2$ into new cluster C with $|C_1| + |C_2|$ elements
    Compute distance from C to all other clusters
    Add a new vertex C to T and connect to vertices $C_1$ and $C_2$
    Remove rows and columns of D corresponding to $C_1$ and $C_2$
    Add a row and column to D corresponding to the new cluster C
  return T
```

Definition of distance between clusters affects clustering!



$\{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_2, g_4, g_6, g_7, g_8, g_9, g_{10}\}$
$\{g_1, g_6, g_7\}$
$\{g_1, g_6\}$
$\{g_2, g_4, g_9, g_{10}\}$
$\{g_2, g_4, g_{10}\}$
$\{g_2, g_4\}$
$\{g_3, g_5, g_8\}$
$\{g_3, g_5\}$
g_3 g_5 g_8 g_7 g_1 g_6 g_{10} g_2 g_4 g_9
8
5
3
9
4
2
10
7
6
1

Hierarchical Clustering – Linkage Criteria



Names	Formula
Maximum or complete-linkage clustering	$\max\{\, d(a,b) : a \in A,\, b \in B \,\}.$
Minimum or single-linkage clustering	$\min\{\, d(a,b) : a \in A,\, b \in B \,\}.$
Mean or average linkage clustering, or UPGMA	$\frac{1}{\lvert A\rvert . \lvert B\rvert} \sum_{a \in A} \sum_{b \in B} d(a,b).$
Centroid linkage clustering, or UPGMC	$\lVert c_s - c_t \rVert$ where c_s and c_t are the centroids of clusters *s* and *t*, respectively.
Minimum energy clustering	$\frac{2}{nm} \sum_{i,j=1}^{n,m} \lVert a_i - b_j \rVert_2 - \frac{1}{n^2} \sum_{i,j=1}^{n} \lVert a_i - a_j \rVert_2 - \frac{1}{m^2} \sum_{i,j=1}^{m} \lVert b_i - b_j \rVert_2$

https://en.wikipedia.org/wiki/Hierarchical_clustering#Linkage_criteria

Outline

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Hierarchical clustering
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

Phylogenetic Tree Reconstruction

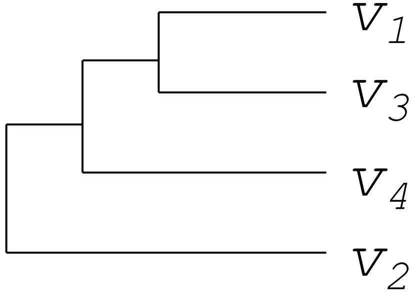
Mouse:	ACAGTGACGCCACACACGT
Gorilla:	CCTGTGACGTAACAAACGA
Chimpanzee:	CCTGTGAGGTAGCAAACGA
Human:	CCTGTGAGGTAGCACACGA

Distance Metric ↓

	v_1	v_2	v_3	v_4
v_1	–			
v_2	.17	–		
v_3	.87	.28	–	
v_4	.59	.33	.62	–

Distance Table

??? →



Phylogenetic Tree

Question: Given sequence data, how to reconstruct tree?

Distance

A **distance** (metric) on a set X is a function $d : X \times X \to \mathbb{R}$ s.t. for all $x, y, z \in X$:

i. $d(x, y) \geq 0$ [non-negativity]

ii. $d(x, y) = 0$ if and only if $x = y$ [identity of indiscernibles]

iii. $d(x, y) = d(y, x)$ [symmetry]

iv. $d(x, y) \leq d(x, z) + d(z, y)$ [triangle inequality]

Examples:

- $X = \mathbb{R}$ and $d(x, y) = |x - y|$
- $X = \Sigma^*$ and d is Hamming distance
- $X = \Sigma^*$ and d is edit distance

Alignment vs. Distance Matrices

```
Mouse:       ACAGTGACGCCACACACGT
Gorilla:     CCTGTGACGTAACAAACGA
Chimpanzee:  CCTGTGAGGTAGCAAACGA
Human:       CCTGTGAGGTAGCACACGA
```

Genes of length m in n species

Easy: use (weighted) edit distance

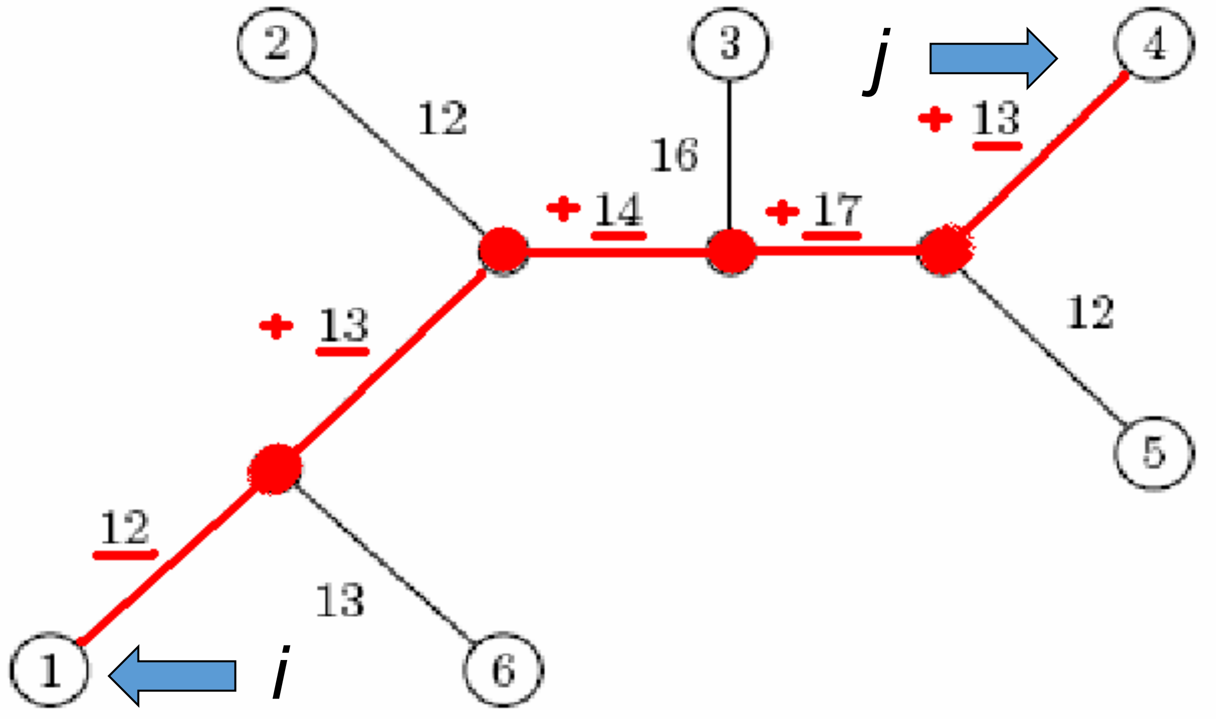
	v_1	v_2	v_3	v_4
v_1	0	7	11	10
v_2	7	0	4	6
v_3	11	4	0	2
v_4	10	6	2	0

$n \times n$ distance matrix

Reverse transformation not possible due to loss of information

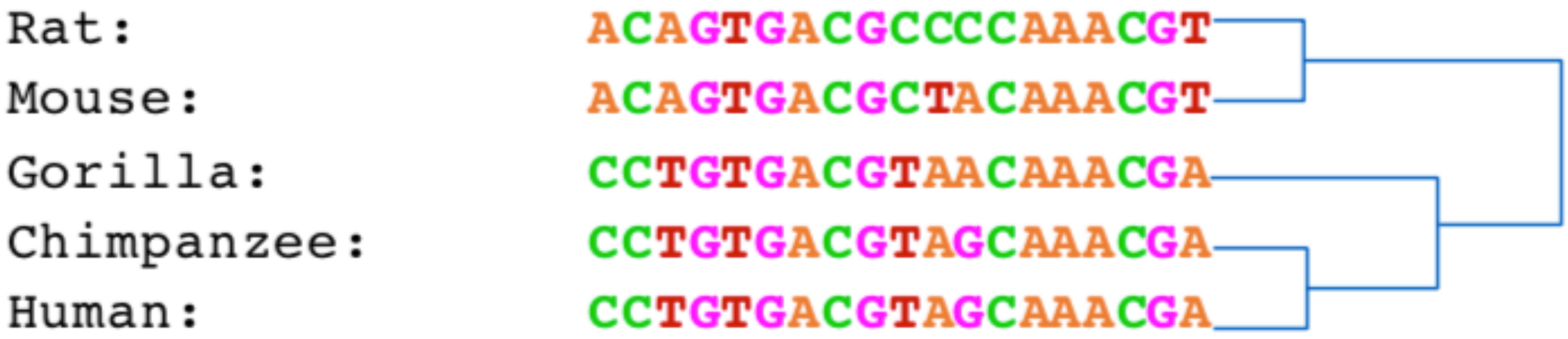
Distances in Trees

Given a tree T with positive edge weights $w(e)$, **tree distance** $d_T(i,j)$ between two leaves i and j is the sum of weights of edges on the unique path from i to j



$d_T(1,4) = 12 + 13 + 14 + 17 + 13 = 69$

General Distance vs. Tree Distance



```
Rat:            ACAGTGACGCCCCAAACGT
Mouse:          ACAGTGACGCTACAAACGT
Gorilla:        CCTGTGACGTAACAAACGA
Chimpanzee:     CCTGTGACGTAGCAAACGA
Human:          CCTGTGACGTAGCAAACGA
```

Tree distance $d_T(i,j)$ not necessarily equal to $d_{i,j}$ as given by distance matrix obtained from alignment

Fitting a Tree to a Given Distance Matrix

- Given n species, we can compute $n \times n$ distance matrix $D = [d_{i,j}]$
- Evolution of these n species is described by an unknown tree
- We need an algorithm to construct tree T that best fits D

Fitting a Tree to a Given Distance Matrix

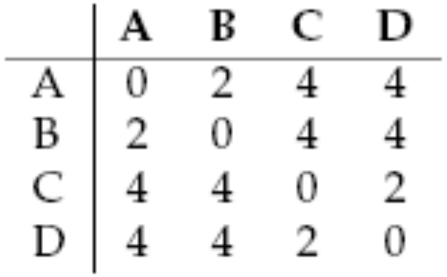
- Given n species, we can compute $n \times n$ distance matrix $D = [d_{i,j}]$
- Evolution of these n species is described by an unknown tree
- We need an algorithm to construct tree T that best fits D

Distance-Based Phylogeny: Given $n \times n$ distance matrix $D = [d_{i,j}]$, find edge-weighted tree T with n leaves that best fits D

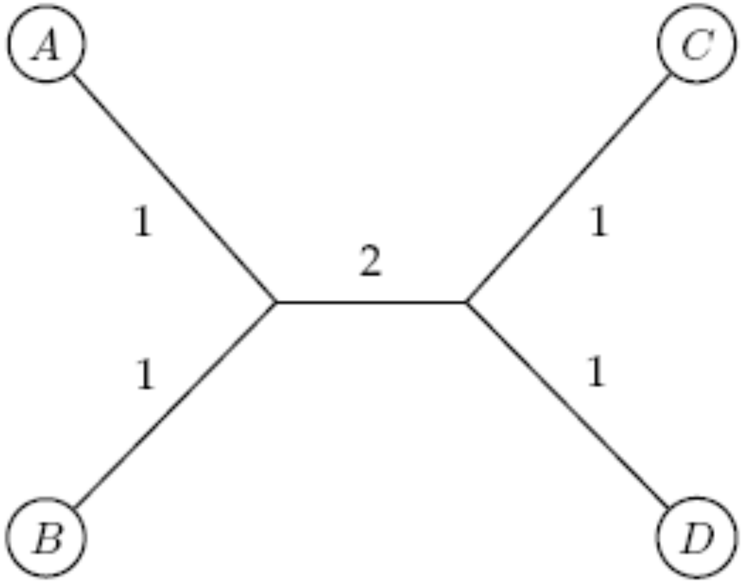
Question: How to define ‘best fit’?

Additive Distance Matrices

Matrix D is ADDITIVE if there exists a tree T with $d_{ij}(T) = D_{ij}$

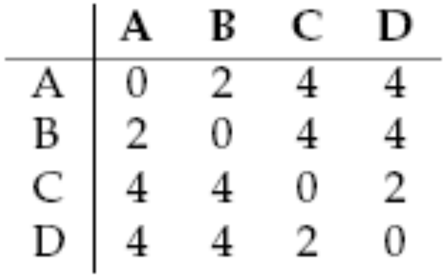


	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0

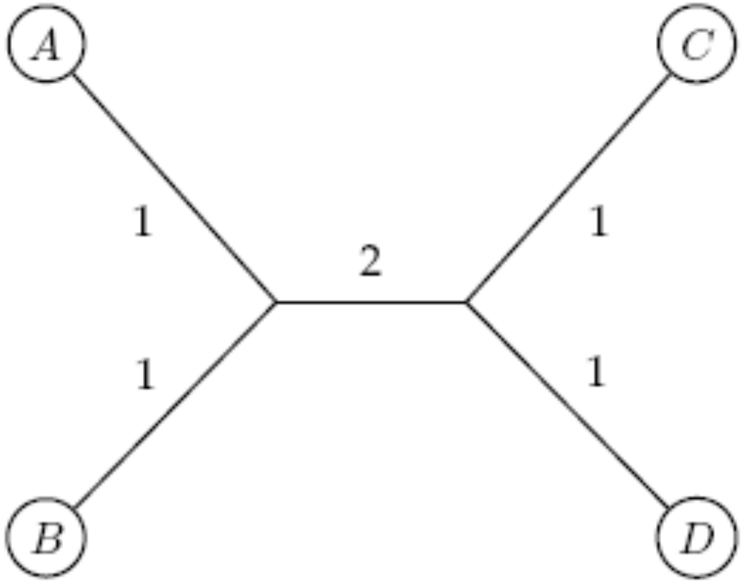


Additive Distance Matrices

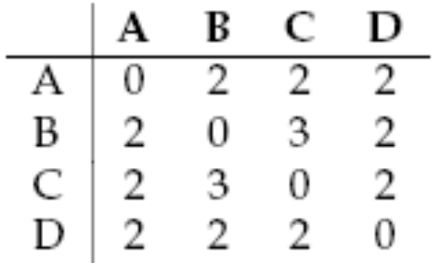
Matrix D is ADDITIVE if there exists a tree T with $d_{ij}(T) = D_{ij}$



	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



NON-ADDITIVE otherwise



	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

?

This is a constructive definition

A Small and a Large Problem

Small Additive Distance Phylogeny Problem:

Given $n \times n$ distance matrix $D = [d_{i,j}]$ and unweighted tree T with n leaves, determine edge weights such that $d_T(i,j) = d_{i,j}$

A Small and a Large Problem

Small Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$ and unweighted tree T with n leaves, determine edge weights such that $d_T(i,j) = d_{i,j}$

Large Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$, find tree T with n leaves **and** edge weights such that $d_T(i,j) = d_{i,j}$

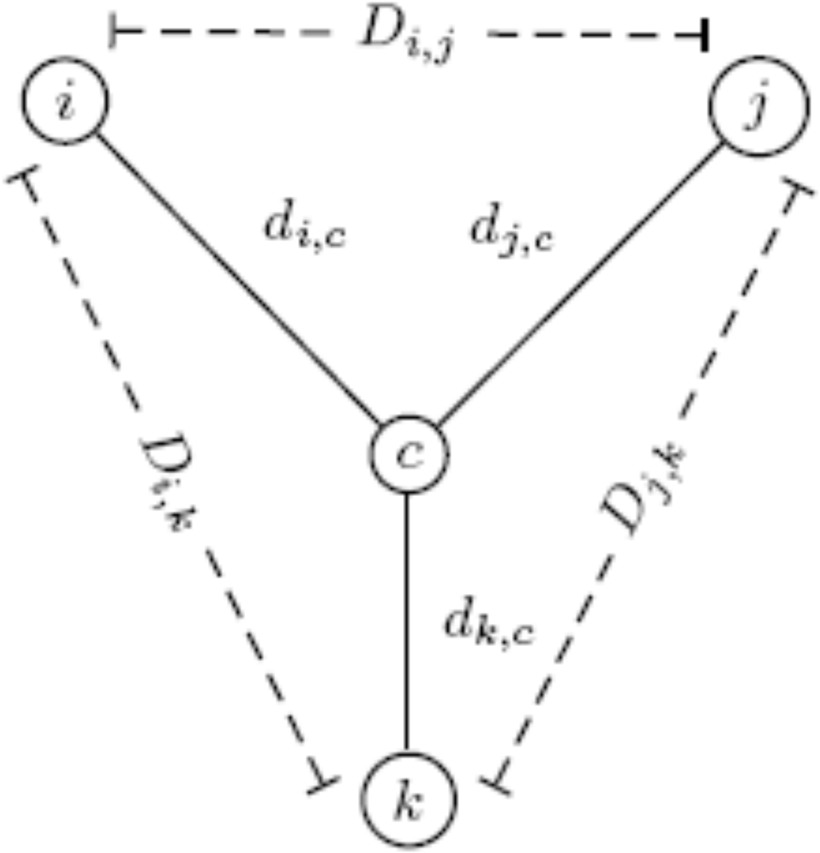
A Small and a Large Problem

Small Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$ and unweighted tree T with n leaves, determine edge weights such that $d_T(i,j) = d_{i,j}$

Large Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$, find tree T with n leaves **and** edge weights such that $d_T(i,j) = d_{i,j}$

Both problems can be solved in polynomial time

Additive Distance Problem with $n = 3$ Sequences

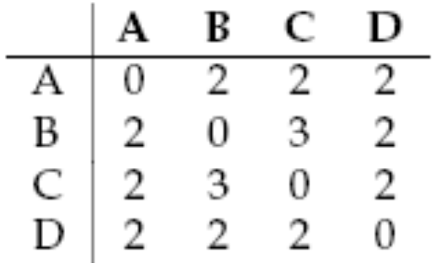


Additive Distance Problem with $n > 3$ Sequences

Unrooted binary tree with n leaves has $2n - 3$ edges and $\binom{n}{2}$ pairwise distances:

- $2n - 3$ variables
- $\binom{n}{2}$ equations

NON-ADDITIVE otherwise →



	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

?

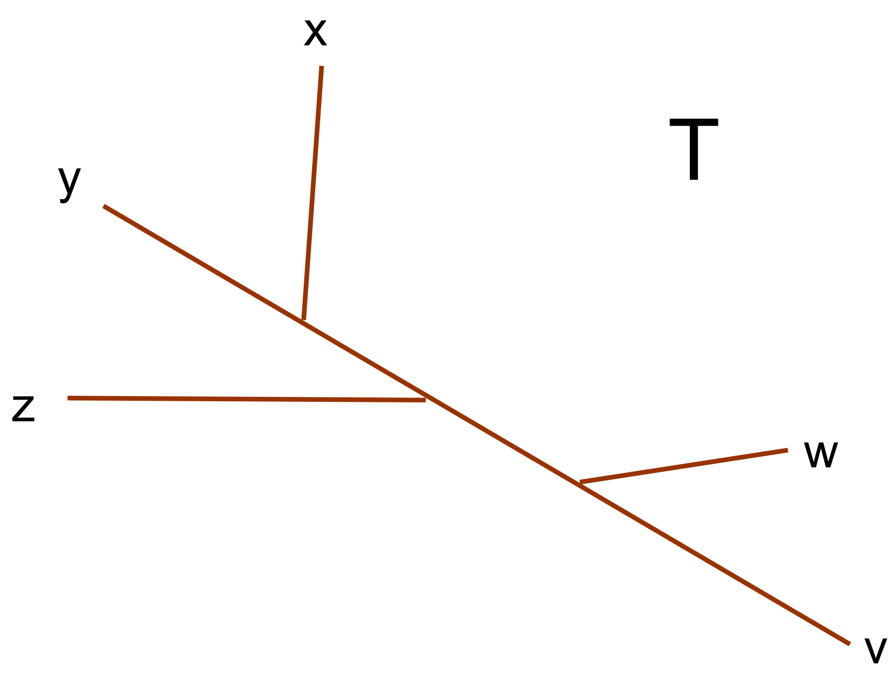
Solution not always possible for $n > 3$

Small Additive Distance Problem

Small Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$ and unweighted tree T with n leaves, determine edge weights such that $d_T(i,j) = d_{i,j}$

D

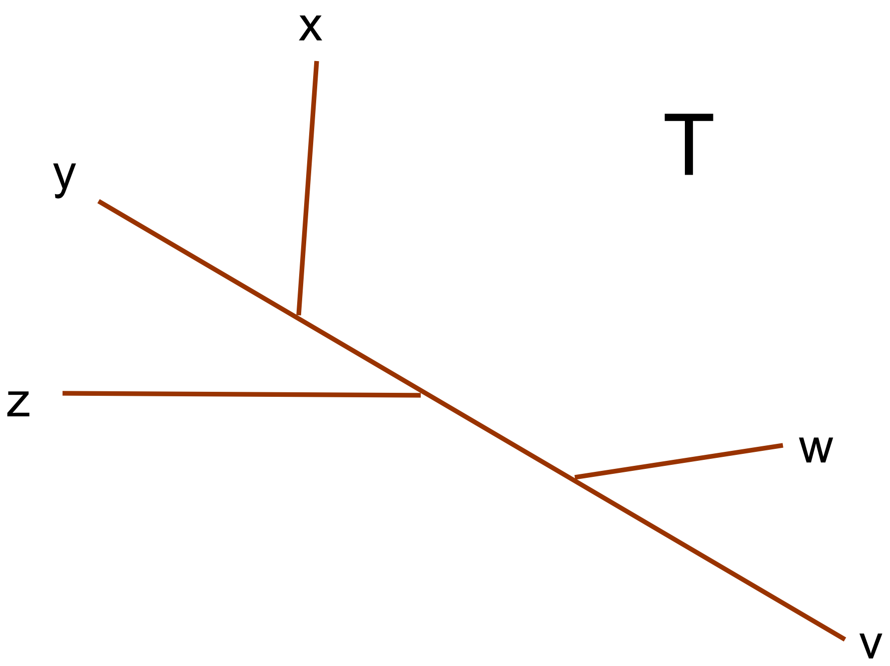
	v	w	x	y	z
v	0	10	17	16	16
w		0	15	14	14
x			0	9	15
y				0	14
z					0



Small Additive Distance Problem

D

	v	w	x	y	z
v	0	10	17	16	16
w		0	15	14	14
x			0	9	15
y				0	14
z					0



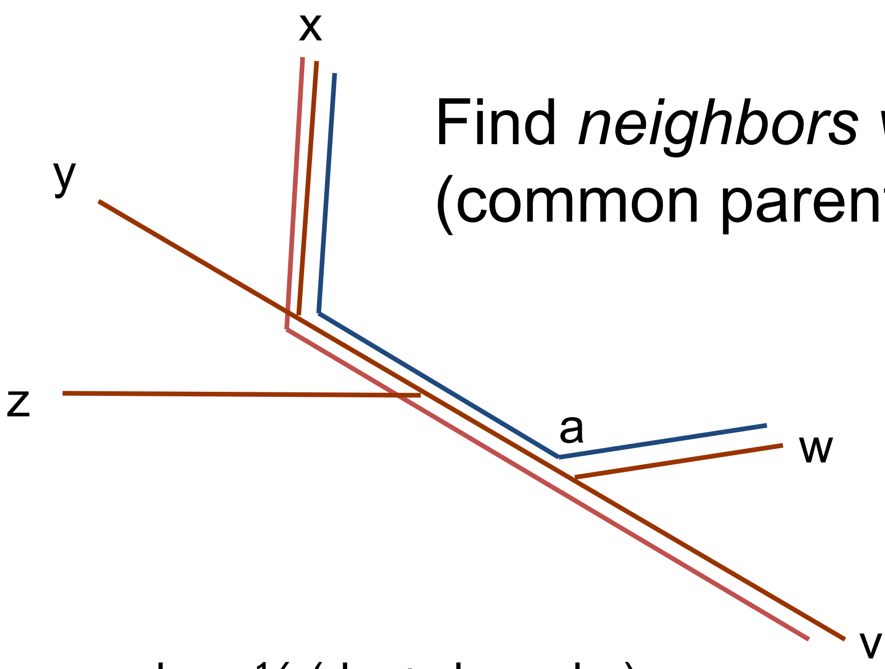
Small Additive Distance Problem

D

	v	w	x	y	z
v	0	10	17	16	16
w		0	15	14	14
x			0	9	15
y				0	14
z					0

D_1

	a	x	y	z
a	0	11	10	10
x		0	9	15
y			0	14
z				0



Find *neighbors v, w* (common parent)

$d_{ax} = ½ (d_{vx} + d_{wx} – d_{vw})$

$d_{ay} = ½ (d_{vy} + d_{wy} – d_{vw})$

$d_{az} = ½ (d_{vz} + d_{wz} – d_{vw})$

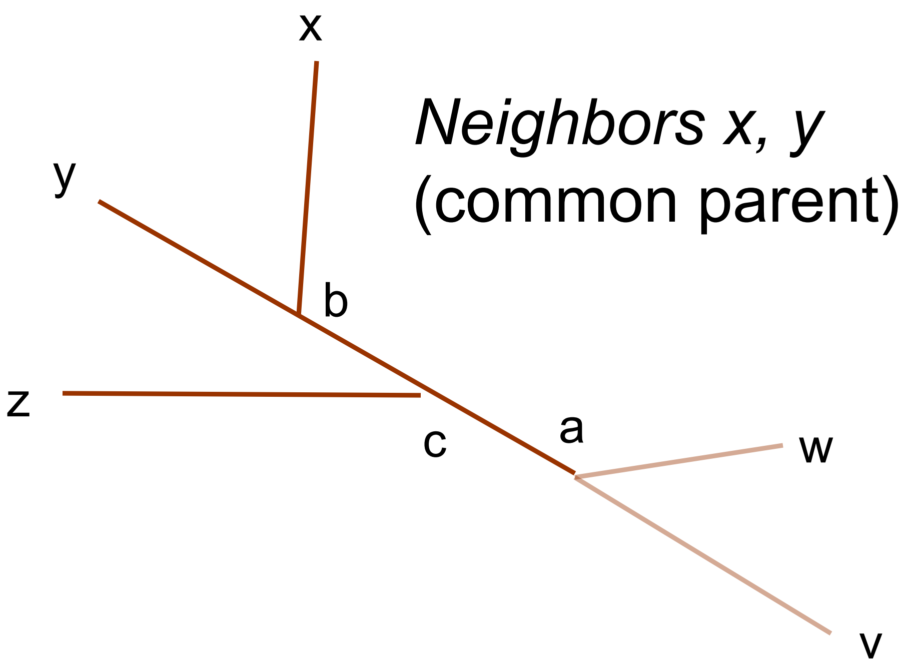
Small Additive Distance Problem

D_1

	a	**x**	**y**	**z**
a	0	11	10	10
x		0	9	15
y			0	14
z				0

D_2

	a	**b**	**z**
a	0	6	10
b		0	10
z			0

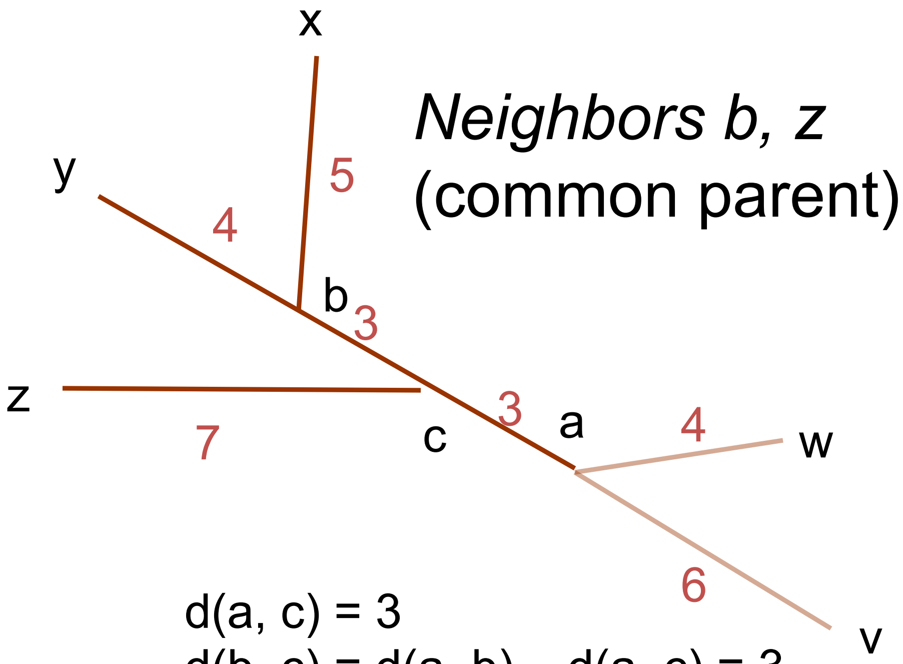


Neighbors x, y
(common parent)

Small Additive Distance Problem

D_1

	a	x	y	z
a	0	11	10	10
x		0	9	15
y			0	14
z				0



Neighbors b, z
(common parent)

D_2

	a	b	z
a	0	6	10
b		0	10
z			0

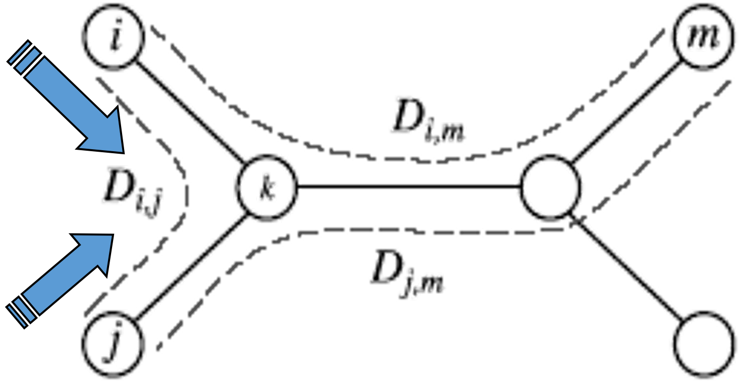
D_3

	a	c
a	0	3
c		0

d(a, c) = 3
d(b, c) = d(a, b) – d(a, c) = 3
d(c, z) = d(a, z) – d(a, c) = 7
d(b, x) = d(a, x) – d(a, b) = 5
d(b, y) = d(a, y) – d(a, b) = 4
d(a, w) = d(z, w) – d(a, z) = 4
d(a, v) = d(z, v) – d(a, z) = 6
Correct!!!

Small Additive Distance Problem

1. Find neighboring leaves i and j with parent k
2. Remove the rows and columns of i and j
3. Add a new row and column corresponding to k, where the distance from k to any other leaf m is computed as



$$d_{k,m} = \frac{(d_{i,m}+d_{j,m}-d_{i,j})}{2}$$

4. Repeat steps 1-3 until tree has only two vertices

A Small and a Large Problem

Small Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$ and unweighted tree T with n leaves, determine edge weights such that $d_T(i,j) = d_{i,j}$

Large Additive Distance Phylogeny Problem:
Given $n \times n$ distance matrix $D = [d_{i,j}]$, find tree T with n leaves **and** edge weights such that $d_T(i,j) = d_{i,j}$

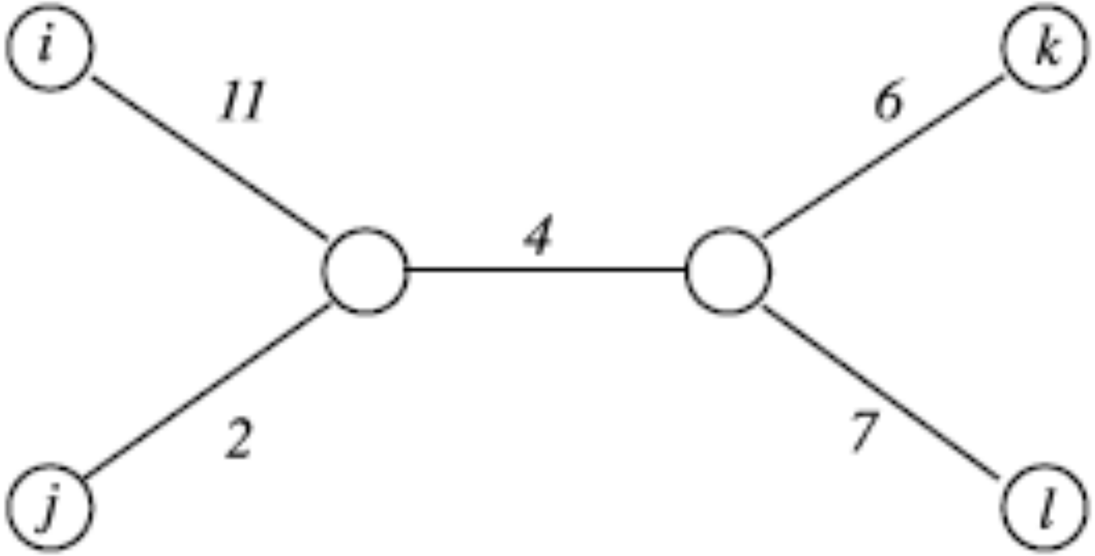
Both problems can be solved in polynomial time

Large Additive Distance Phylogeny Problem

Idea: find neighboring leaves by simply selecting pair of closest leaves

WRONG!

	i	***j***	***k***	***l***
i	0	13	21	22
j		0	12	13
k			0	13
l				0



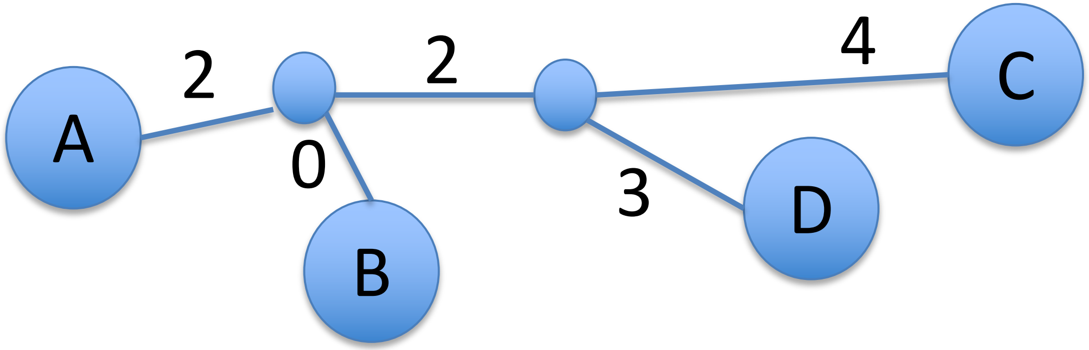
i and *j* are neighbors, but (d_{ij} = 13) > (d_{jk} = 12).

Finding a pair of neighboring leaves is a nontrivial problem!

Degenerate Triples

A **degenerate triple** is a set of three distinct elements $i, j, k \in [n]$ such that $d_{i,j} + d_{j,k} = d_{i,k}$

	A	B	C	D
A	0	2	8	7
B		0	6	5
C			0	7
D				0



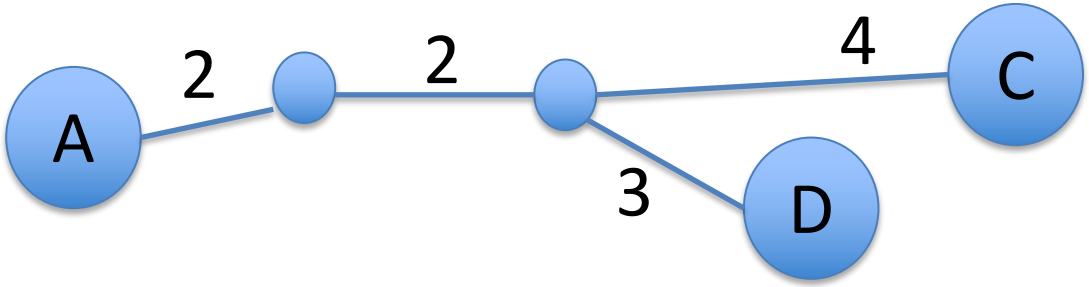
Element j in a degenerate triple (i, j, k) lies* on the evolutionary path from i to k

*or is attached to this path by an edge of length 0

Degenerate Triples can be Removed

A **degenerate triple** is a set of three distinct elements $i, j, k \in [n]$ such that $d_{i,j} + d_{j,k} = d_{i,k}$

	A	C	D
A	0	8	7
C		0	7
D			0

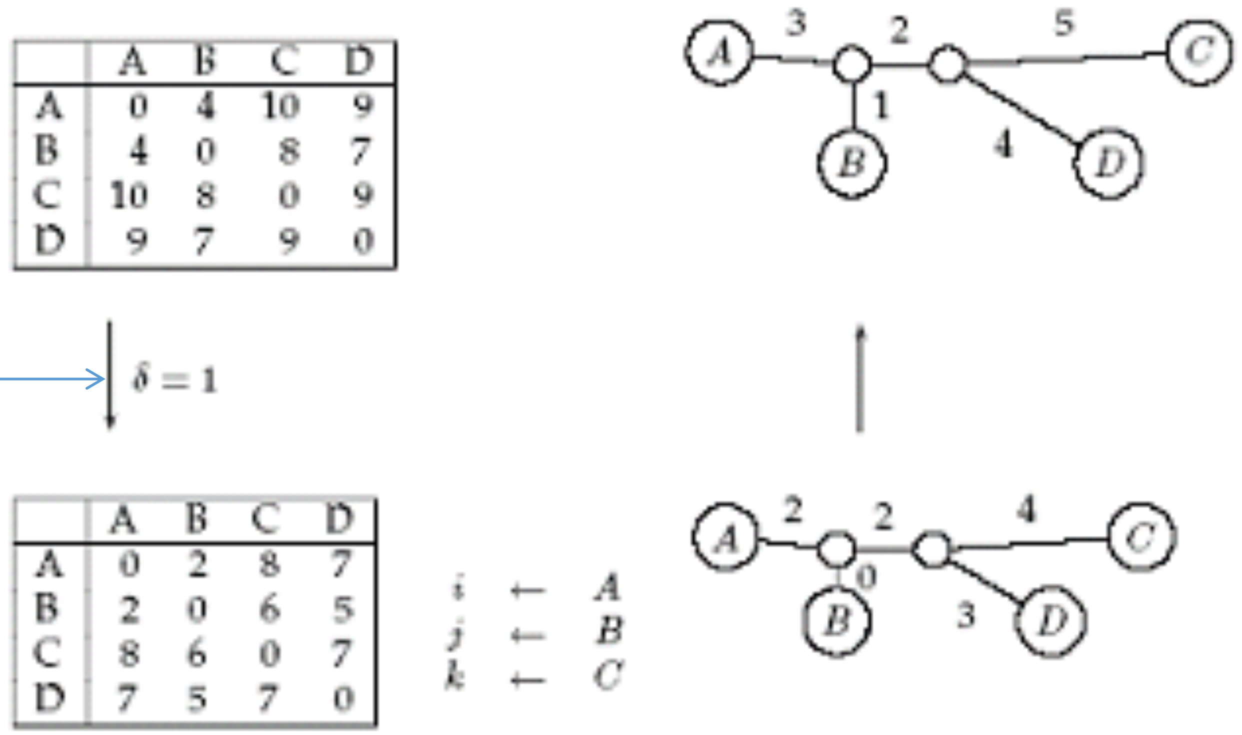


Element j in a degenerate triple (i, j, k) lies* on the evolutionary path from i to k

*or is attached to this path by an edge of length 0

Looking for Degenerate Triples

If distance matrix D does not have a degenerate triple, one can create one by shortening all hanging edges



	A	B	C	D
A	0	4	10	9
B	4	0	8	7
C	10	8	0	9
D	9	7	9	0

Trimming Parameter

$\delta = 1$

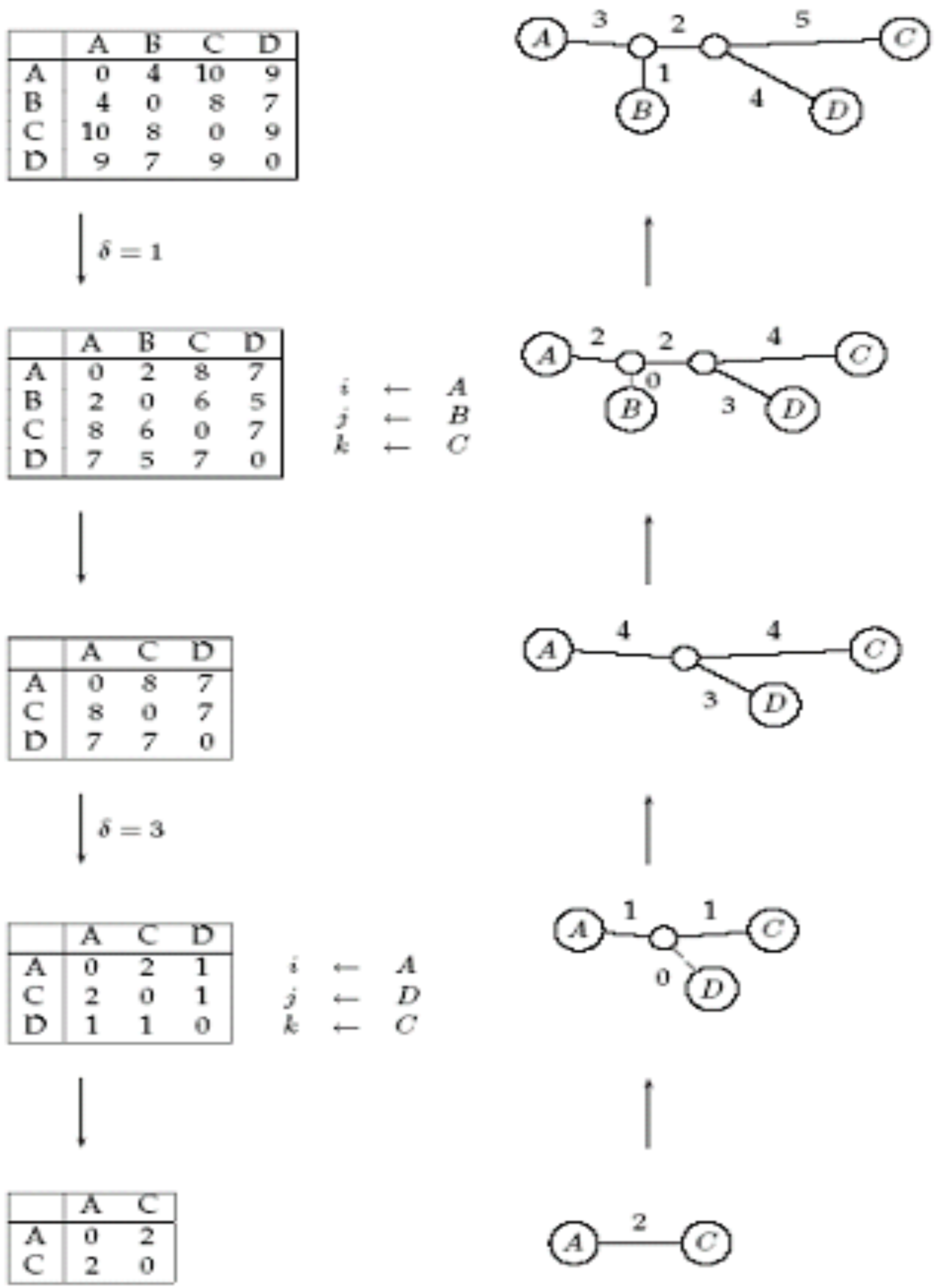
	A	B	C	D
A	0	2	8	7
B	2	0	6	5
C	8	6	0	7
D	7	5	7	0

$i \leftarrow A$
$j \leftarrow B$
$k \leftarrow C$

Decrease entries in matrix D by 2δ

Additive Phylogeny

- If there is no degenerative triple:
  - Reduce all hanging edges by the same amount δ, so that all pairwise distances in the matrix are reduced by 2δ.
- This process will eventually collapse one of the leaves (when δ equals the length of the shortest hanging edge), forming a degenerate triple (i, j, k) and reducing the size of the distance matrix D
- The attachment point for j can be recovered in the reverse transformations by saving $d_{i,j}$ for each collapsed leaf.



	A	B	C	D
A	0	4	10	9
B	4	0	8	7
C	10	8	0	9
D	9	7	9	0

$\delta = 1$

	A	B	C	D
A	0	2	8	7
B	2	0	6	5
C	8	6	0	7
D	7	5	7	0

$i \leftarrow A$
$j \leftarrow B$
$k \leftarrow C$

	A	C	D
A	0	8	7
C	8	0	7
D	7	7	0

$\delta = 3$

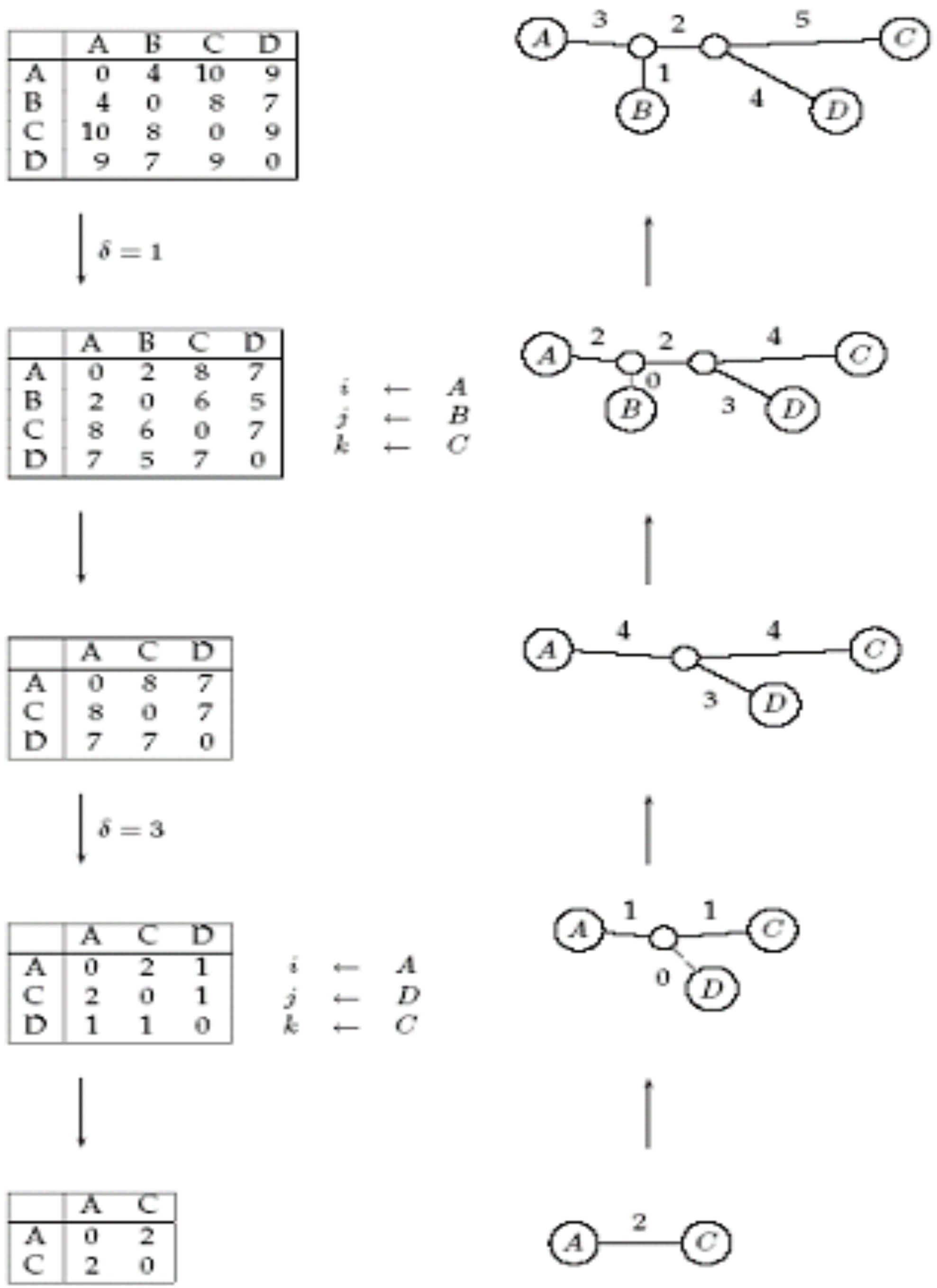
	A	C	D
A	0	2	1
C	2	0	1
D	1	1	0

$i \leftarrow A$
$j \leftarrow D$
$k \leftarrow C$

	A	C
A	0	2
C	2	0

Additive Phylogeny

```
AdditivePhylogeny(D)
  if D is a 2 x 2 matrix
    T = tree of a single edge of length $D_{1,2}$
    return T
  if D is non-degenerate
    Compute trimming parameter $\delta$
    Trim(D, $\delta$)
  Find a triple i, j, k in D such that $D_{ij} + D_{jk} = D_{ik}$
  $x = D_{ij}$
  Remove $j^{th}$ row and $j^{th}$ column from D
  T = AdditivePhylogeny(D).
  Add a new vertex v to T at distance x from i to k
  Add j back to T by creating an edge (v,j) of
      length 0
  for every leaf l in T
    if distance from l to v in the tree ≠ $D_{l,j}$
      output "matrix is not additive"
      return
  Extend all "hanging" edges by length $\delta$
  return T
```



	A	B	C	D
A	0	4	10	9
B	4	0	8	7
C	10	8	0	9
D	9	7	9	0

$\delta = 1$

	A	B	C	D
A	0	2	8	7
B	2	0	6	5
C	8	6	0	7
D	7	5	7	0

$i \leftarrow A$
$j \leftarrow B$
$k \leftarrow C$

	A	C	D
A	0	8	7
C	8	0	7
D	7	7	0

$\delta = 3$

	A	C	D
A	0	2	1
C	2	0	1
D	1	1	0

$i \leftarrow A$
$j \leftarrow D$
$k \leftarrow C$

	A	C
A	0	2
C	2	0

Outline

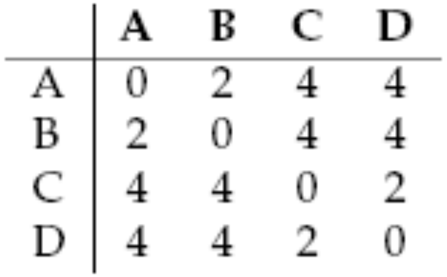
- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

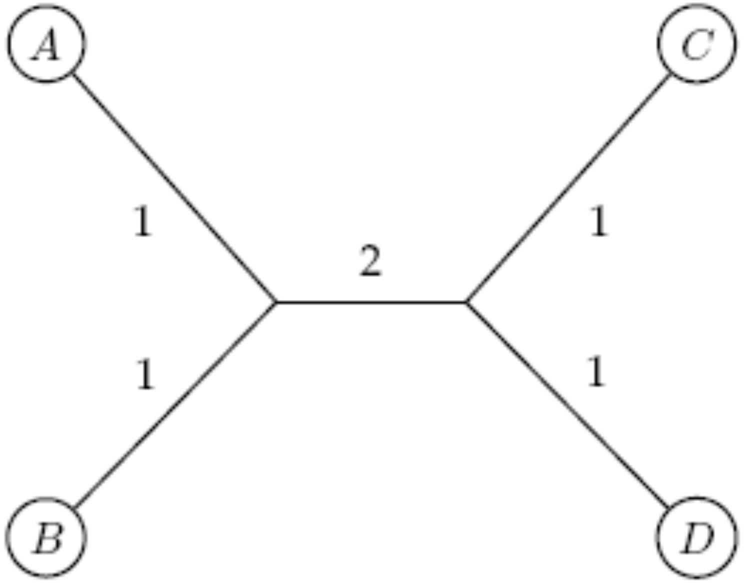
- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

Additive Distance Matrices

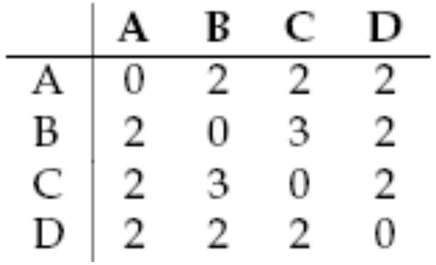
Matrix *D* is ADDITIVE if there exists a tree *T* with $d_{ij}(T) = D_{ij}$



	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



NON-ADDITIVE otherwise



	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

?

This is a constructive definition

Question: Can we characterize set of additive matrices?

Four Point Condition (Zaretskii 1965, Buneman 1971)

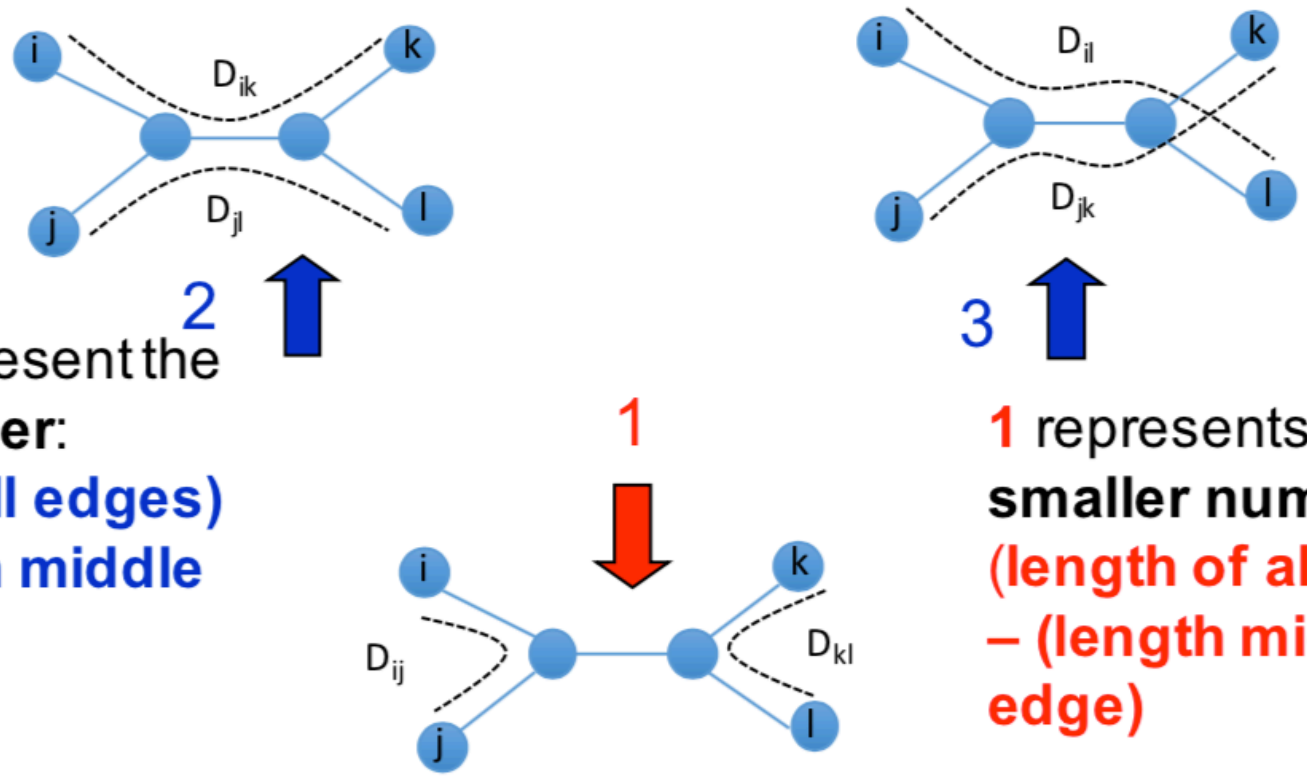
Four point condition of matrix $D = [d_{i,j}]$:

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Three sums:

1. $d_{i,j} + d_{k,l}$
2. $d_{i,k} + d_{j,l}$
3. $d_{i,l} + d_{j,k}$



2 and **3** represent the **same number**: **(length of all edges) + 2 * (length middle edge)**

1 represents a **smaller number**: **(length of all edges) – (length middle edge)**

Four Point Condition

Four point condition of matrix $D = [d_{i,j}]$:

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

If two leaves are the same, four point condition is triangle inequality (e.g. set $l = j$)

Four point condition generalizes triangle inequality and defines a subset of distances, namely additive distances

Four Point Condition: Theorem

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Theorem: An $n \times n$ matrix D is additive if and only if the for point condition holds for every quartet $(i, j, k, l) \in [n]^4$

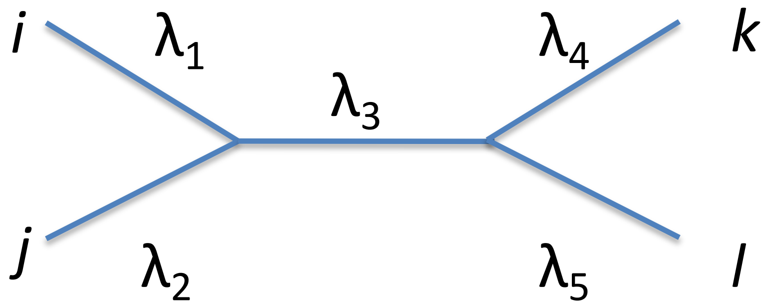
Four Point Condition: Theorem

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Theorem: An $n \times n$ matrix D is additive if and only if the for point condition holds for every quartet $(i, j, k, l) \in [n]^4$

Proof: (=>) Since D is additive, there is a tree T such that $d_{i,j} = d_T(i, j)$ for all $(i, j) \in n^2$. Let (i, j, k, l) be a quartet. Assume w.l.o.g. that i, j and k, l are neighbors. Define λ_m as illustrated.



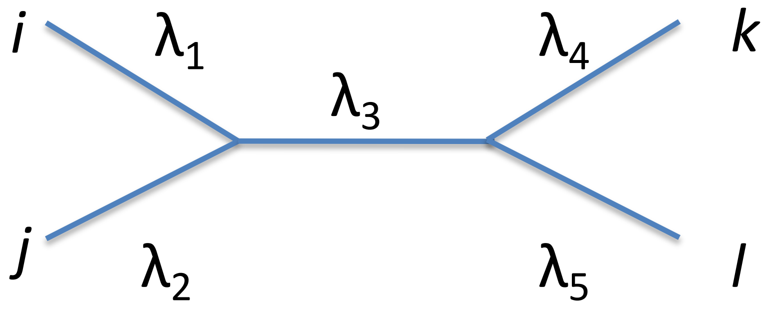
Four Point Condition: Theorem

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Theorem: An $n \times n$ matrix D is additive if and only if the for point condition holds for every quartet $(i, j, k, l) \in [n]^4$

Proof: (=>) Since D is additive, there is a tree T such that $d_{i,j} = d_T(i, j)$ for all $(i, j) \in n^2$. Let (i, j, k, l) be a quartet. Assume w.l.o.g. that i, j and k, l are neighbors. Define λ_m as illustrated.



$$d_{i,k} + d_{j,l} = (\lambda_1 + \lambda_3 + \lambda_4) + (\lambda_2 + \lambda_3 + \lambda_5) = d_{i,l} + d_{j,k}$$
$$\geq (\lambda_1 + \lambda_2) + (\lambda_4 + \lambda_5) = d_{i,j} + d_{k,l}$$

Four Point Condition: Theorem

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Theorem: An $n \times n$ matrix D is additive if and only if the for point condition holds for every quartet $(i, j, k, l) \in [n]^4$

Proof: (<=) Assume four point condition holds. Need an algorithm to construct T. AdditivePhylogeny(T) is one such algorithm*. Neighbor joining is another algorithm.

*we have not proved correctness nor shown how to correct δ

Additive Distance Matrix

Four point condition of matrix $D = [d_{i,j}]$:

Every four leaves (quartet) can be labeled as (i, j, k, l) such that

$$d_{i,j} + d_{k,l} \leq d_{i,k} + d_{j,l} = d_{i,l} + d_{j,k}$$

Theorem: Let D be an $n \times n$ matrix. The following statements are equivalent.

1. Matrix D is additive.
2. There exists a unique tree T (modulo isomorphism) s.t. $d_{i,j} = d_T(i, j)$ for all $(i, j) \in n^2$.
3. Four point condition holds for every quartet $(i, j, k, l) \in [n]^4$.

Outline

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner

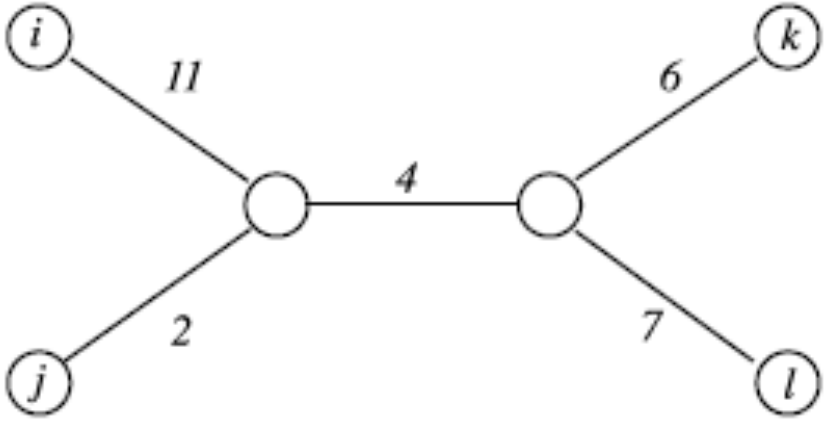
Distance Based Phylogeny Problem

Large Additive Distance Phylogeny Problem:
Given $n \times n$ matrix $D = [d_{i,j}]$, find tree T with n leaves **and** edge weights such that $\max_{(i,j)\in[n]^2} |d_T(i,j) - d_{i,j}|$ is minimum.

Equivalently, find additive matrix D' closest to input matrix D

Neighbor Joining Algorithm (Saitou and Nei 1987)

- Constructs binary unrooted trees.
- Recall: leaves a and b are neighbors if they have a common parent
- Recall: closest leaves are not necessarily neighbors
- NJ: Find pair of leaves that are “close” to each other but “far” from other leaves

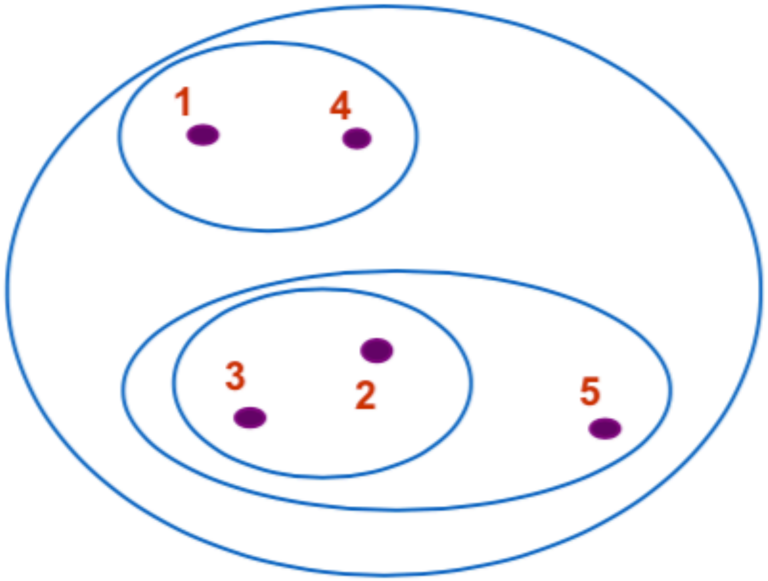


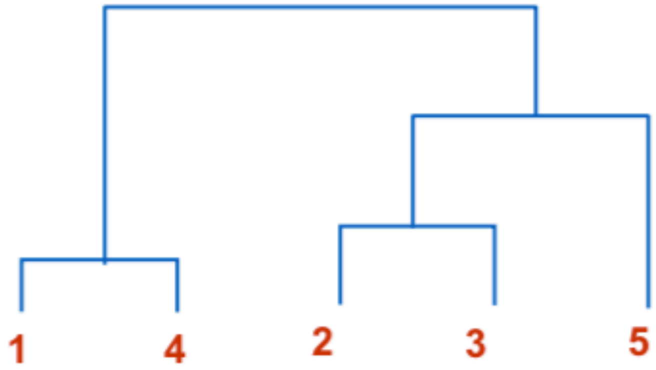
Two advantages: (1) reproduces correct tree for additive matrix, and (2) otherwise gives good approximation of correct tree

Distance Trees as Hierarchical Clustering

Leaves = Data points.

Data points clustered/grouped into hierarchy according to some distance criterion.

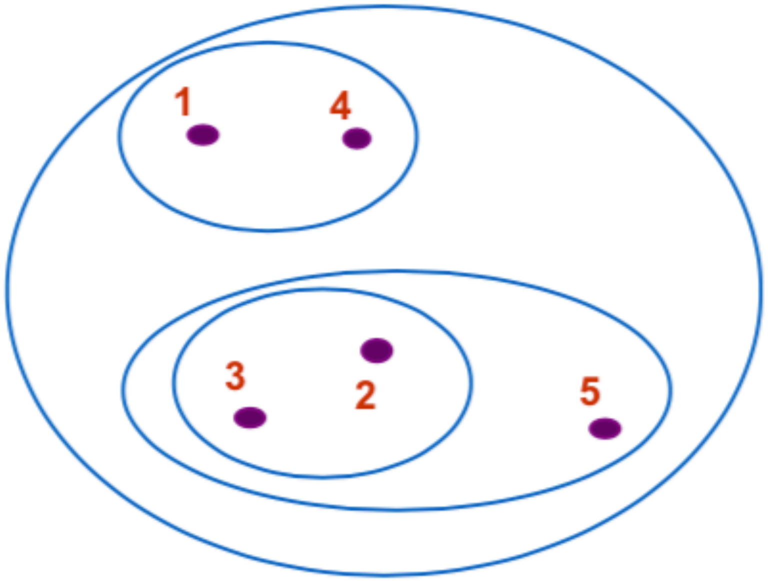


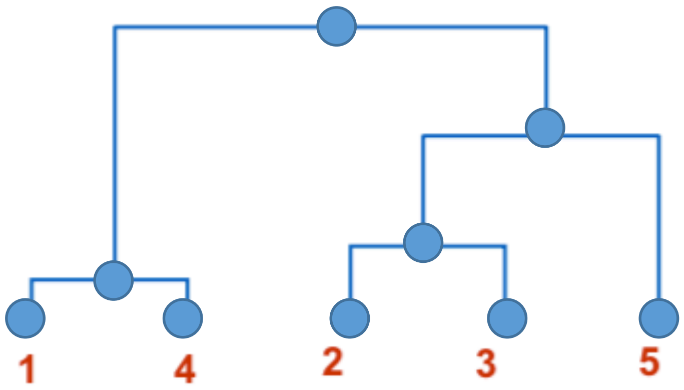


Distance Trees as Hierarchical Clustering

Leaves = Data points.

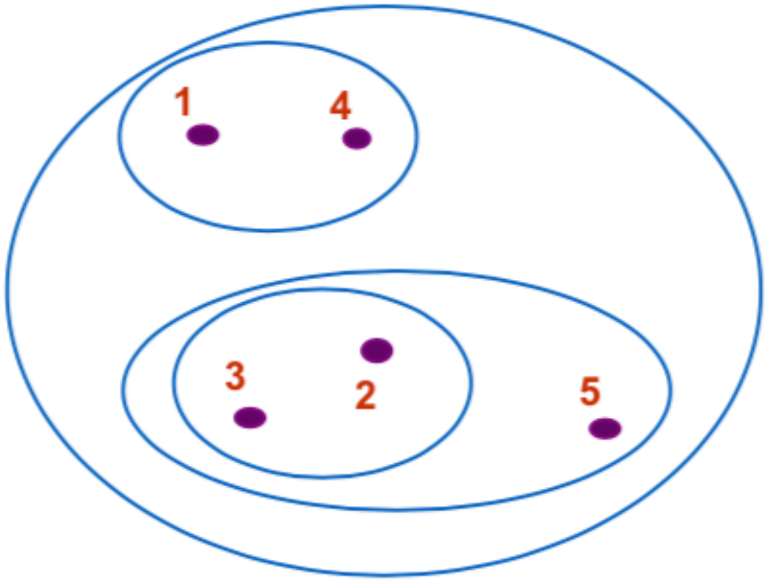
Data points clustered/grouped into hierarchy according to some distance criterion.



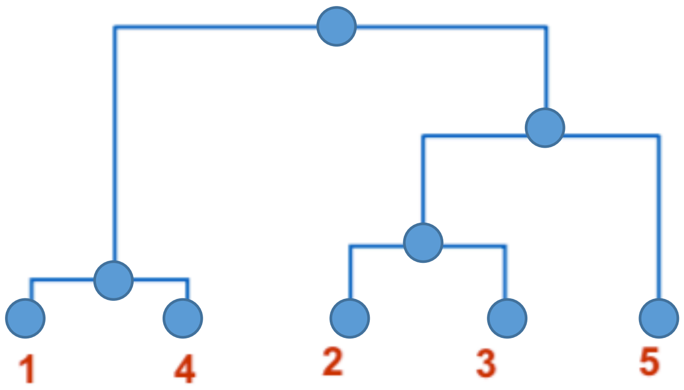


Distance Trees as Hierarchical Clustering

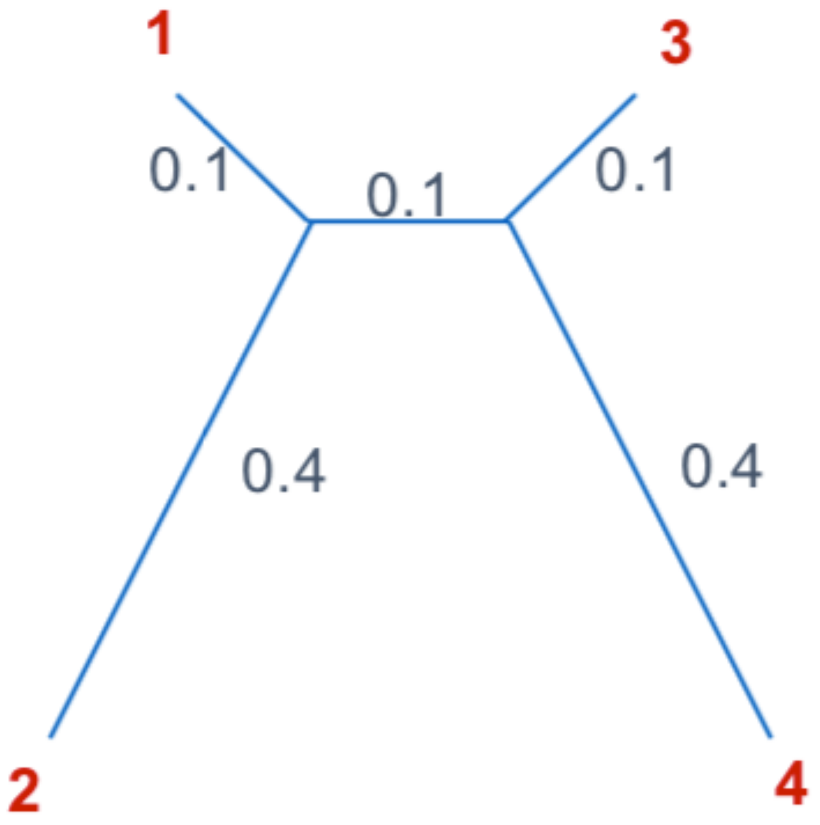
```
Hierarchical Clustering ($D$ , $n$)
  Form $n$ clusters each with one element
  Construct a graph $T$ by assigning one vertex to each cluster
  while there is more than one cluster
    Find the two closest clusters $C_1$ and $C_2$
    Merge $C_1$ and $C_2$ into new cluster $C$ with $|C_1| + |C_2|$ elements
    Compute distance from $C$ to all other clusters
    Add a new vertex $C$ to $T$ and connect to vertices $C_1$ and $C_2$
    Remove rows and columns of $D$ corresponding to $C_1$ and $C_2$
    Add a row and column to $D$ corresponding to the new cluster $C$
  return $T$
```



Selection criterion: distance between clusters affects clustering!



Neighbor Joining: Selection Criterion



Let ***C*** = {1, …, n} be current clusters/leaves.

Define: $u_i = \sum_k D(i, k)$.

Intuitively, u_i measures separation of *i* from other leaves.

Goal: Minimize D(*i*, *j*) and maximize $u_i + u_j$.

Solution: Find pair (i, j) that minimizes:

$$S_D(i, j) = (n - 2)\, D(i, j) - u_i - u_j$$

Claim: Given additive matrix D.

$S_D(x, y) = \min S_D(i, j)$ if and only if *x* and *y* are neighbors in tree T with $d_T = D$.

Neighboring Joining: Algorithm

Initialization:

Form n clusters $C_1, C_2, \ldots, C_n$, one for each leaf node.

Define tree T to be the set of leaf nodes, one per sequence.

Iteration: (D is $m \times m$)

Pick i, j such that $S_D(i, j) = (m - 2)\,D(i, j) - u_i - u_j$ is minimal.

Merge i and j into new node $[ij]$ in T.

Assign length ½ $(D(i, j) + 1/(m\text{-}2)\ (u_i - u_j))$ to edge $(i, [ij])$

Assign length ½ $(\ D(i, j) + 1/(m\text{-}2)\ (u_j - u_i))$ to edge $(j, [ij])$

Remove rows and columns from D corresponding to i and j.

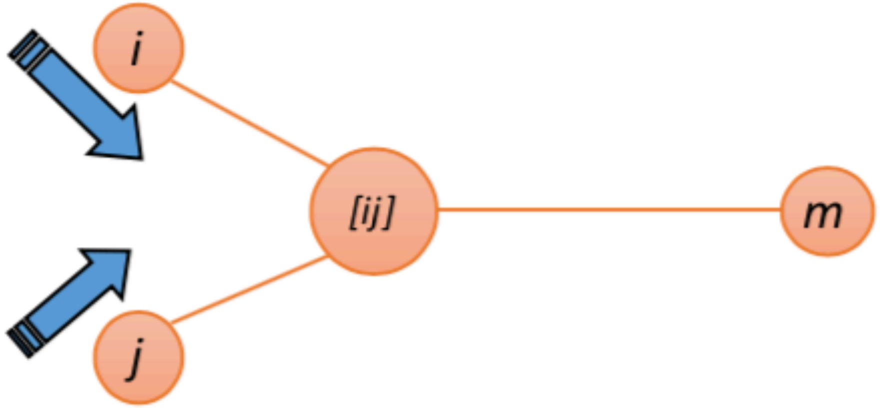
Add row and column to D for new vertex $[ij]$.

Set $D([ij], m) = ½\,[\ D(i, m) + D(j, m) - D(i,j)]$

Temination:

When only one cluster

Question: Does this create rooted or unrooted trees?



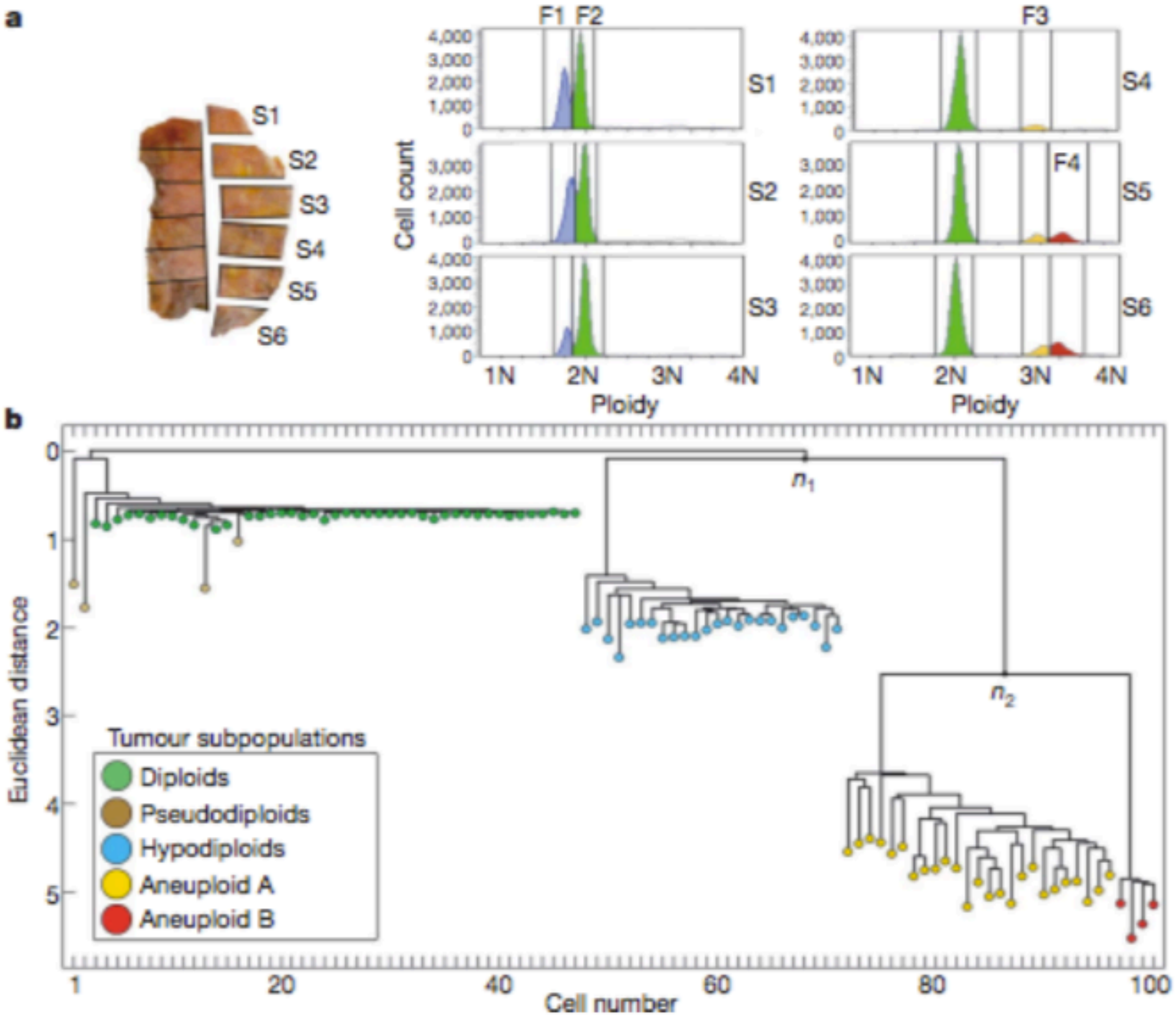
Advantages of Neighbor Joining

Theorem: Let D be an $n \times n$ matrix. If matrix D is additive then neighbor joining produces the unique phylogenetic tree T (modulo isomorphism) such that $d_{i,j} = d_T(i,j)$ for all $(i,j) \in n^2$.

Theorem: Let D be an $n \times n$ matrix. If there exists an additive matrix D' such that $|D - D'|_\infty \leq 0.5$ then neighbor joining applied to D reconstructs the unique tree T (modulo isomorphism) such that $d'_{i,j} = d_T(i,j)$ for all $(i,j) \in n^2$.

Atteson 1991

Neighbor Joining in Practice



[Navin et al, Nature 2011]

Neighbor Joining tree relating copy number profiles from single cells in a tumor.

Summary

- Recap: RNA Secondary Structure Prediction
- Phylogenetics introduction
- Hierarchical clustering
- Additive distance phylogeny
- Four point condition
- Neighbor joining

Reading:

- Chapter 10.2 and 10.5-10.8 in Jones and Pevzner