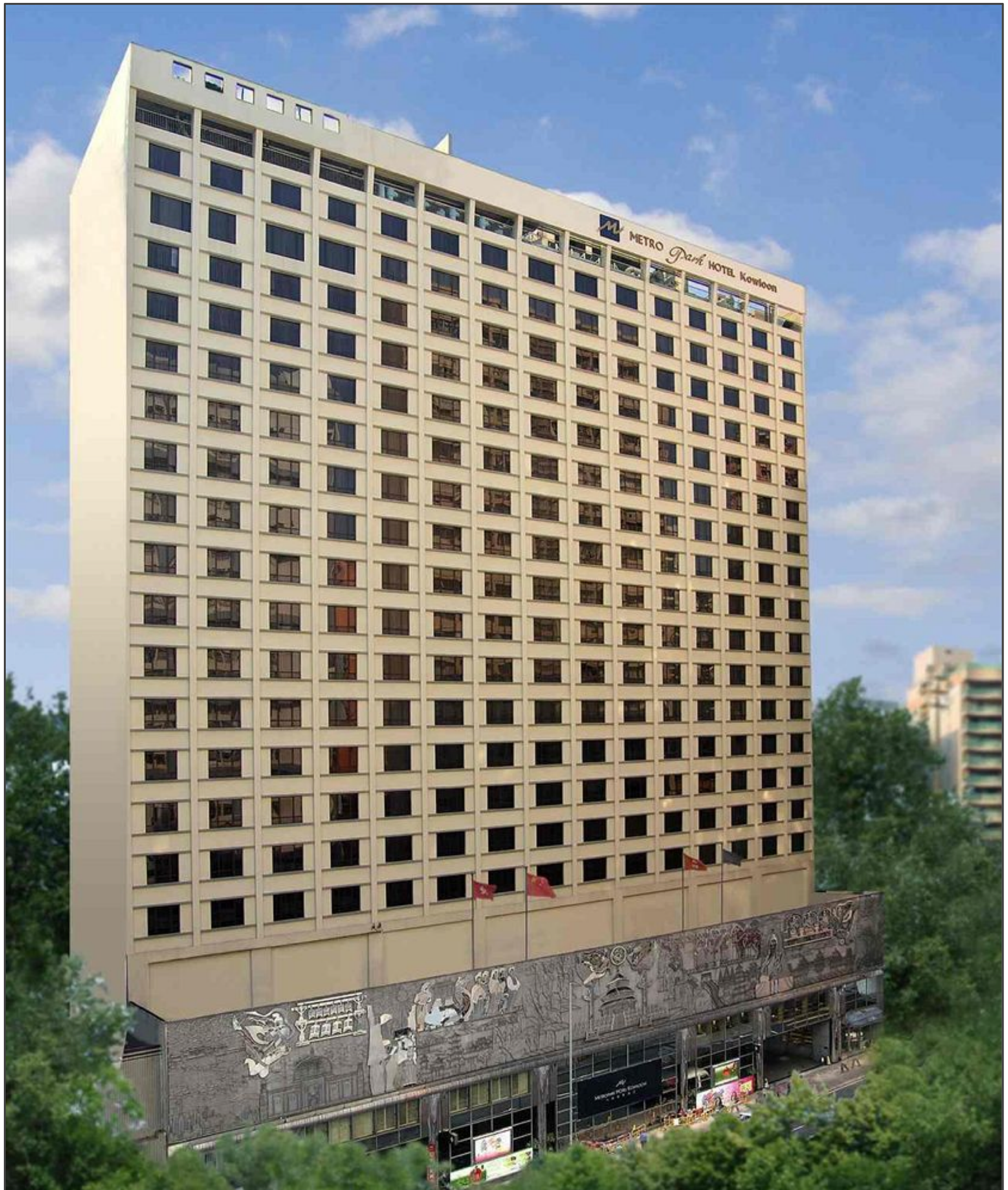
Which Animal Gave Us SARS?

Evolutionary Trees Part 1:
The Neighbor-Joining Algorithm

Phillip Compeau and Pavel Pevzner

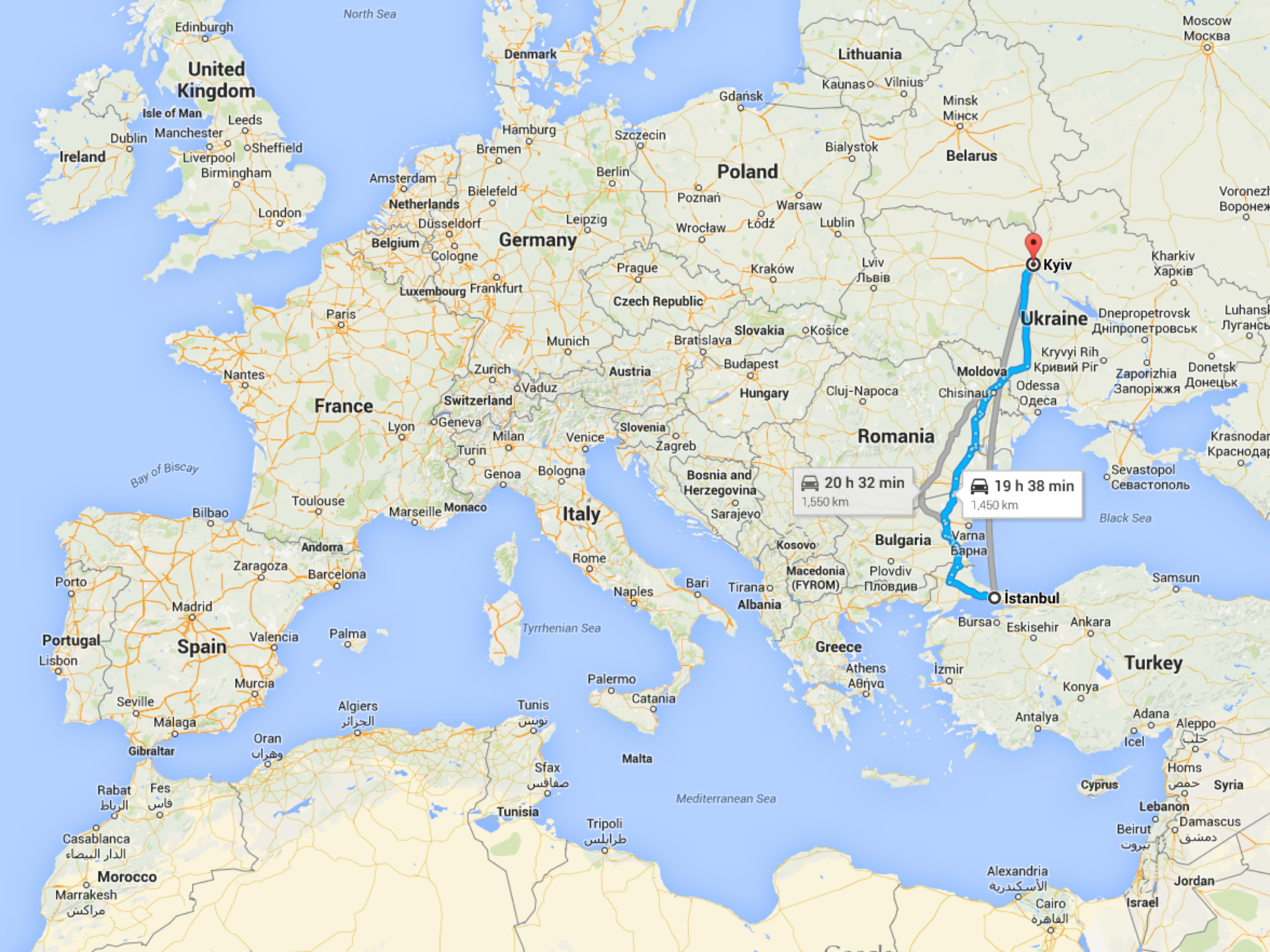
Bioinformatics Algorithms: An Active Learning Approach

©2018 by Compeau and Pevzner. All rights reserved.



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

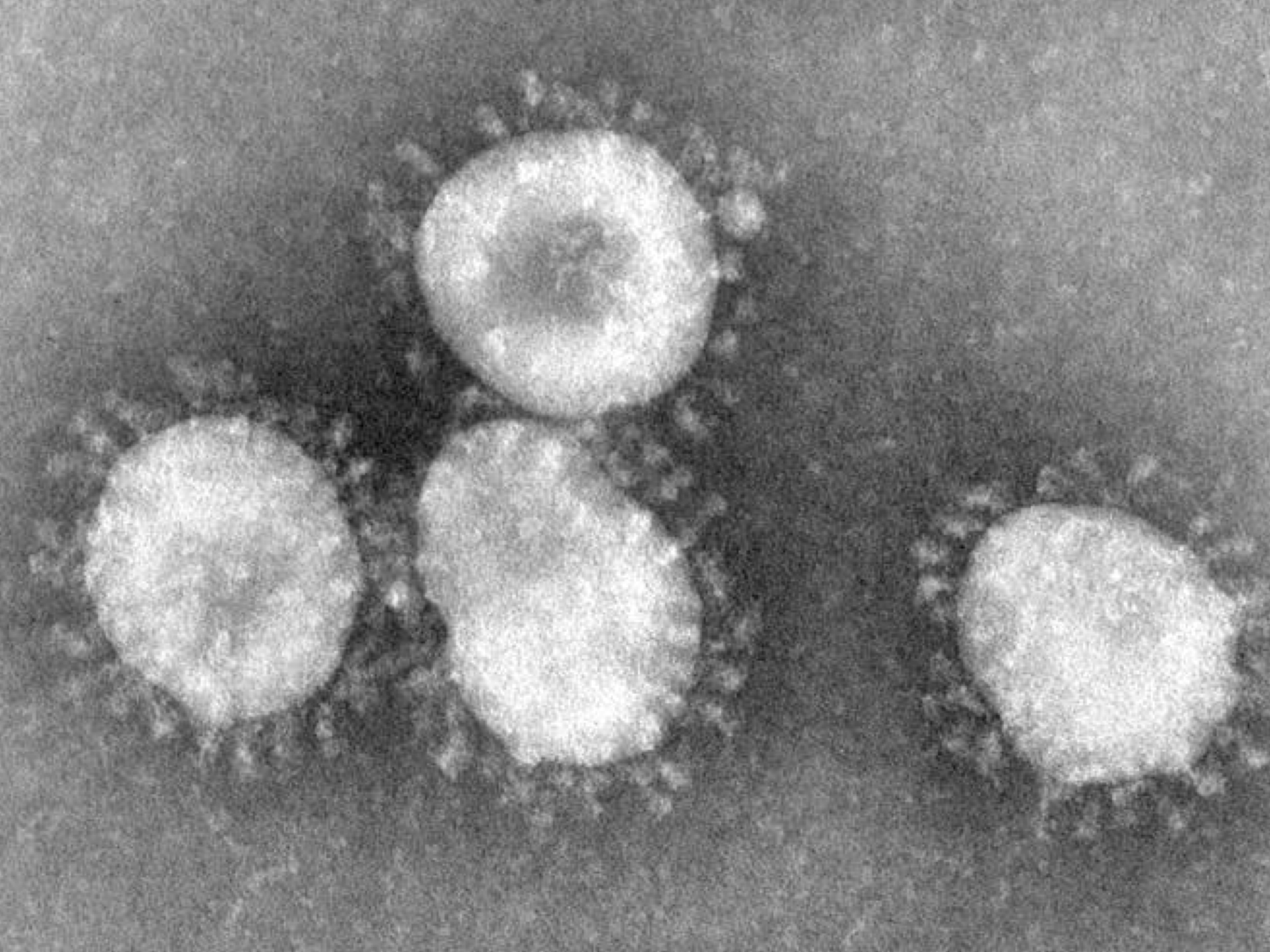

Modern boundaries are
shown for reference.
Cities with repeated plague outbreaks, 14th to 18th century
Spreading of the plague by marine routes
Approximate date of the first outbreak
1347
1350
1348
1351
1349
relatively unaffected
London
Amsterdam
Rotterdam
Antwerp
Aachen
Lübeck
Danzig (Gdansk)
Kiev
Paris
Strassburg (Strasbourg)
Nürnberg
Vienna
approaching from Asia, 1346
Bordeaux
Lyon
Florence
Venice
Trieste
Genoa
Marseille
Lisbon
Barcelona
Ancona
Ragusa (Dubrovnik)
Valencia
Rome
Seville
Naples
Constantinople (Istanbul)
approaching from Asia, 1346
© 2012 Encyclopædia Britannica, Inc.



20 h 32 min
1,550 km
19 h 38 min
1,450 km
Kyiv
İstanbul
Ukraine
Moldova
Chisinau
Odessa
Одеса
Varna
Варна
Bulgaria
Romania
Black Sea
North Sea
Mediterranean Sea
Tyrrhenian Sea
Bay of Biscay
United Kingdom
Ireland
Isle of Man
Edinburgh
Leeds
Dublin
Manchester
Sheffield
Liverpool
Birmingham
London
Denmark
Lithuania
Kaunas
Vilnius
Minsk
Мінск
Belarus
Moscow
Москва
Gdańsk
Hamburg
Szczecin
Bialystok
Bremen
Berlin
Poland
Amsterdam
Bielefeld
Poznań
Warsaw
Netherlands
Düsseldorf
Leipzig
Wrocław
Łódź
Lublin
Belgium
Cologne
Germany
Voronezh
Воронеж
Prague
Kraków
Lviv
Львів
Kharkiv
Харків
Luxembourg
Frankfurt
Czech Republic
Paris
Slovakia
Košice
Dnepropetrovsk
Днiпропетровськ
Luhansk
Луганськ
Munich
Bratislava
Kryvyi Rih
Кривий Ріг
Nantes
Zurich
Austria
Budapest
Zaporizhia
Запоріжжя
Donetsk
Донецьк
Vaduz
Hungary
Cluj-Napoca
France
Switzerland
Lyon
Geneva
Milan
Venice
Slovenia
Zagreb
Krasnodar
Краснодар
Turin
Sevastopol
Севастополь
Genoa
Bologna
Bosnia and Herzegovina
Sarajevo
Toulouse
Marseille
Monaco
Italy
Bilbao
Andorra
Kosovo
Rome
Zaragoza
Barcelona
Macedonia (FYROM)
Plovdiv
Пловдив
Samsun
Porto
Madrid
Bari
Tirana
Albania
Naples
Bursa
Eskisehir
Ankara
Portugal
Valencia
Palma
Spain
Greece
Lisbon
Athens
Αθήνα
İzmir
Turkey
Murcia
Palermo
Konya
Seville
Catania
Algiers
الجزائر
Tunis
تونس
Antalya
Adana
Aleppo
حلب
Málaga
Icel
Gibraltar
Oran
وهران
Malta
Sfax
صفاقس
Homs
حمص
Syria
Cyprus
Rabat
الرباط
Fes
فاس
Tunisia
Lebanon
Damascus
دمشق
Beirut
بيروت
Casablanca
الدار البيضاء
Tripoli
طرابلس
Morocco
Alexandria
الأسكندرية
Jordan
Marrakesh
مراكش
Cairo
القاهرة
Israel


Ireland
1 case
China
Canada
29 cases
United States
1 case
Guangdong
Province
Metropole
Hotel
Hong Kong
Special
Administrative
Region
195 cases
Vietnam
58 cases
Singapore
71 cases
Philippine
Sea
South
China
Sea
Arabian
Sea
Bay
of
Bengal
Indian
The Spread
of ?????

The Spread of SARS



Questions about SARS

Which animal gave us SARS? How does SARS compare to other viruses and how did it mutate over time?

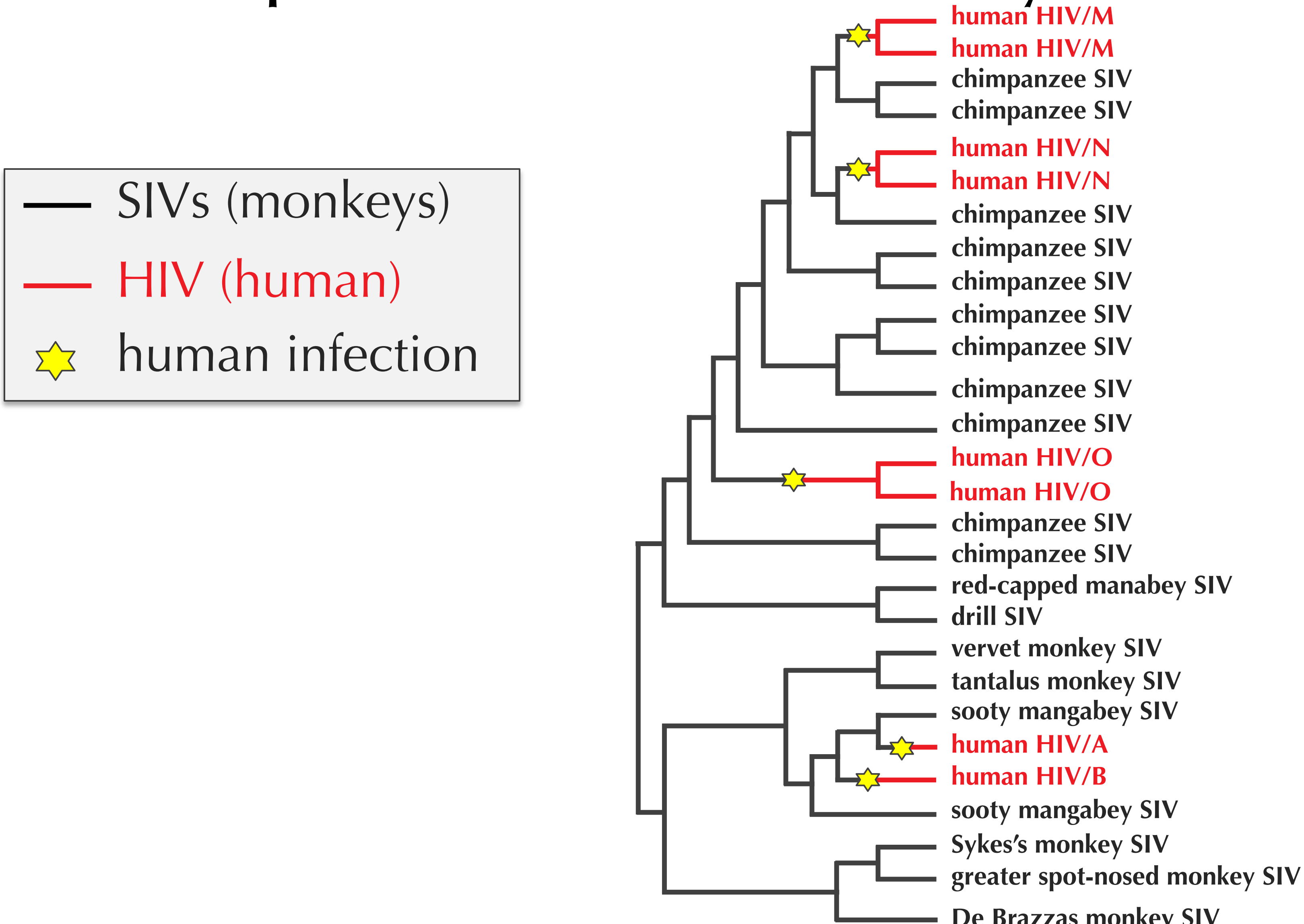
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Questions about SARS

Which animal gave us SARS? How does SARS compare to other viruses and how did it mutate over time?

To answer these questions, we need to learn how to construct **evolutionary trees** (a.k.a. **phylogenies**).

Example: HIV Evolutionary Tree

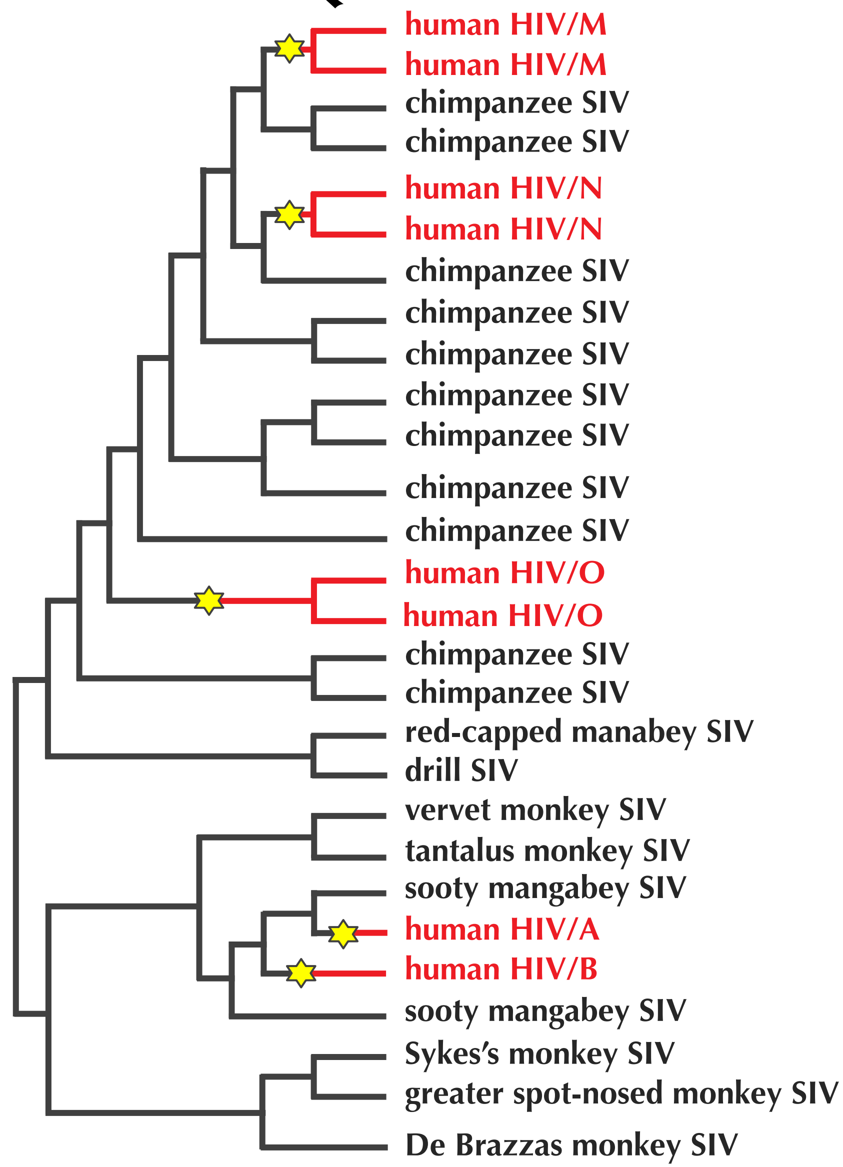


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Two Computational Questions

How do we construct the tree's *structure*?

Can we infer anything about the ancestors?



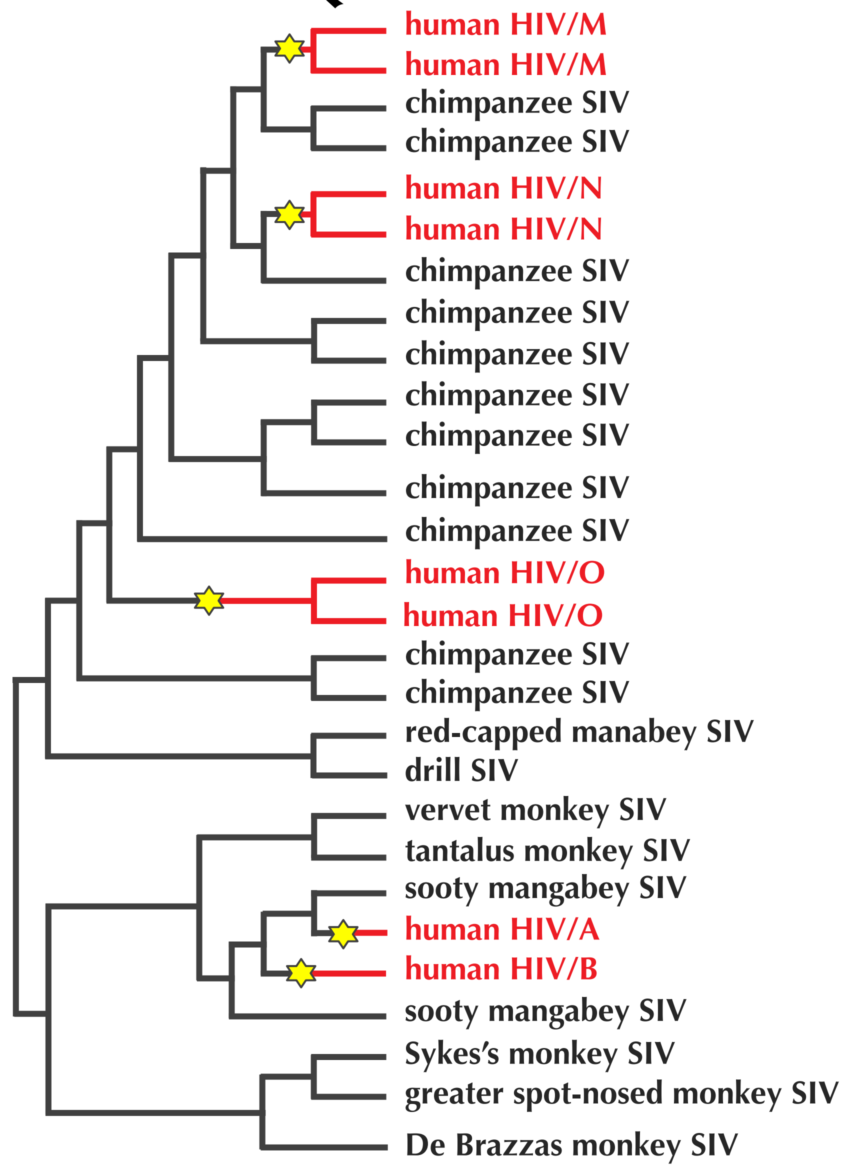
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Two Computational Questions

How do we construct the tree's *structure*?

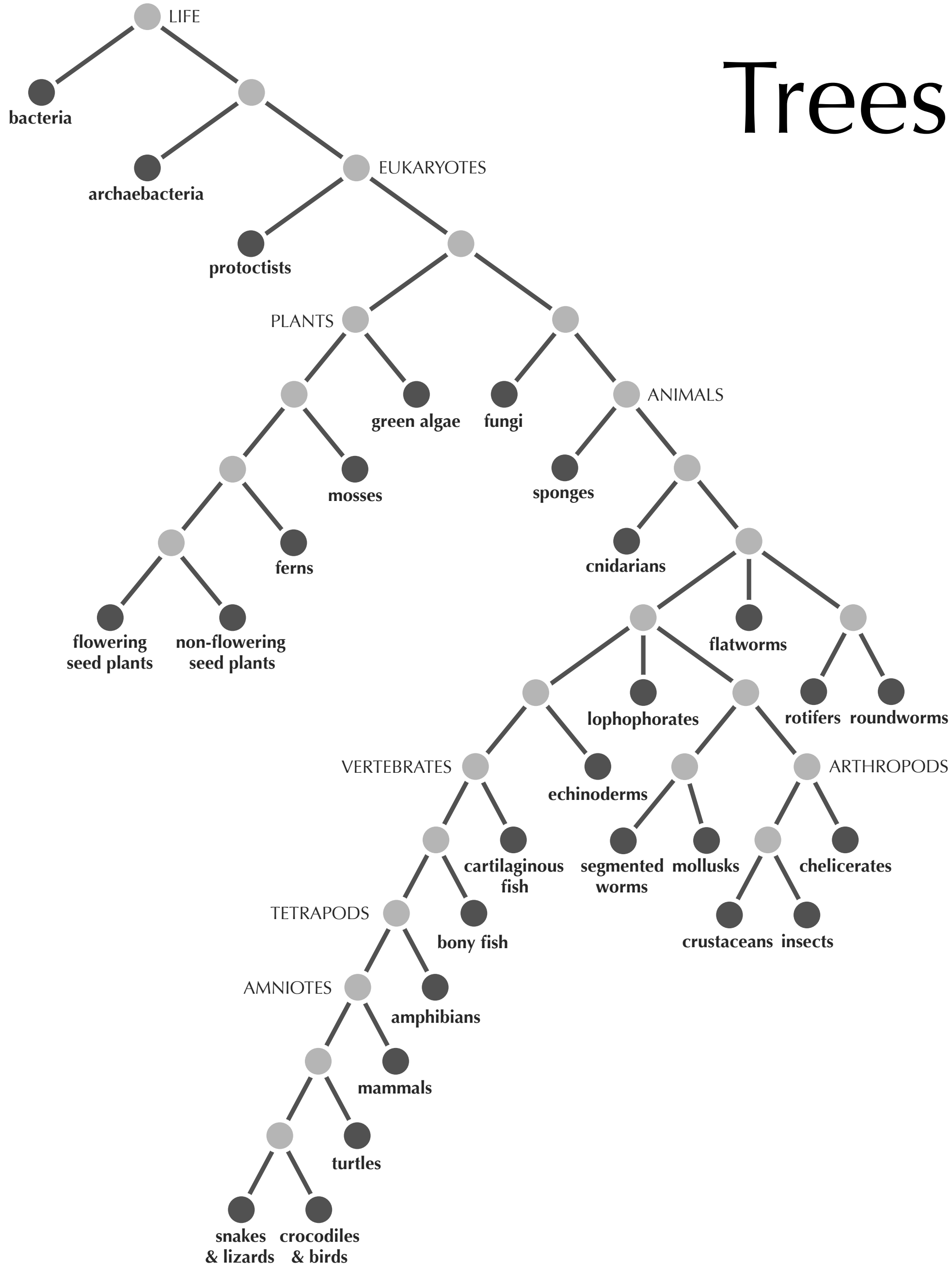
Can we infer anything about the ancestors?

Checkpoint: Any thoughts on how we could answer either question?



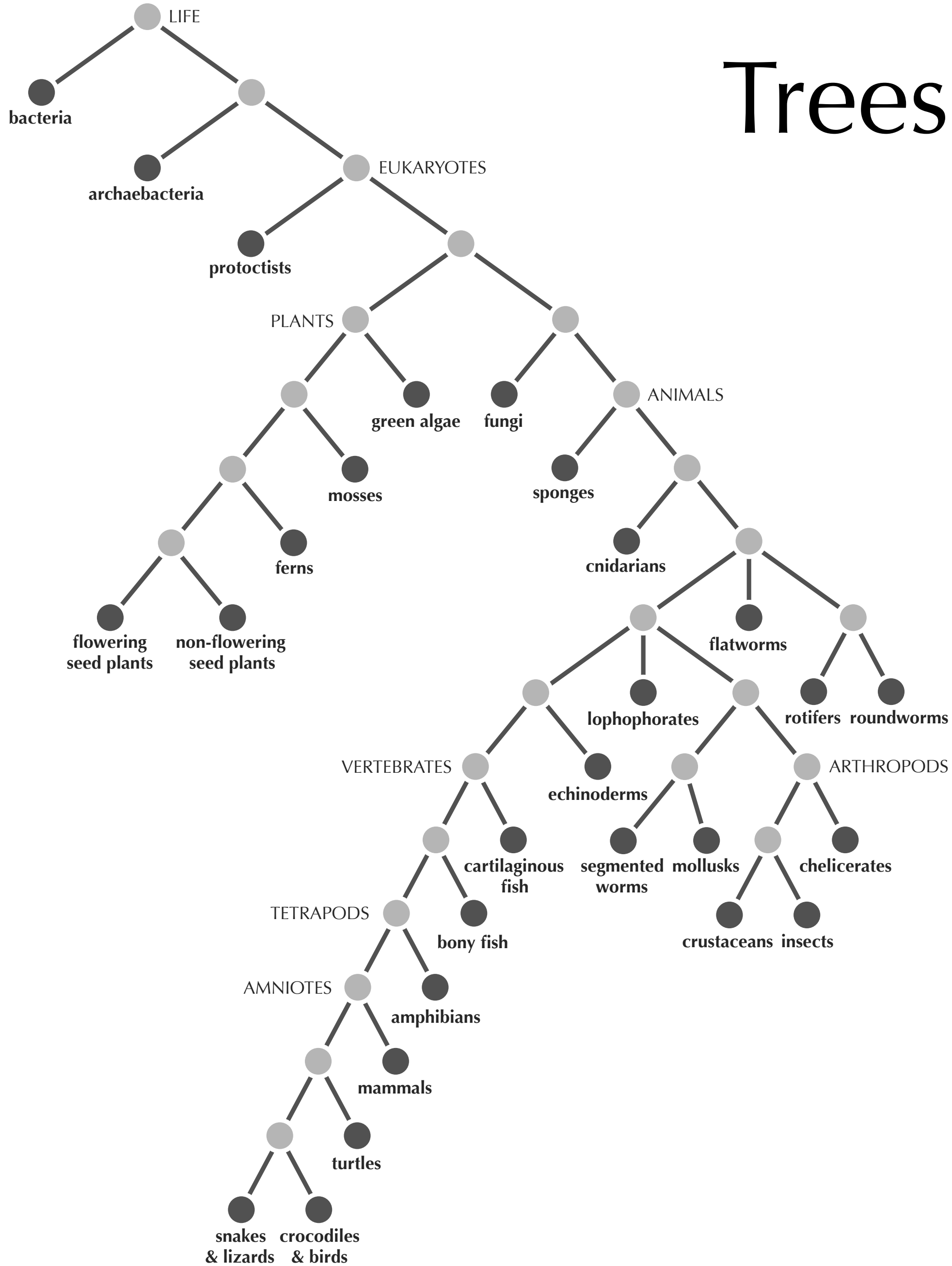
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Trees



Tree: Connected graph containing no cycles.

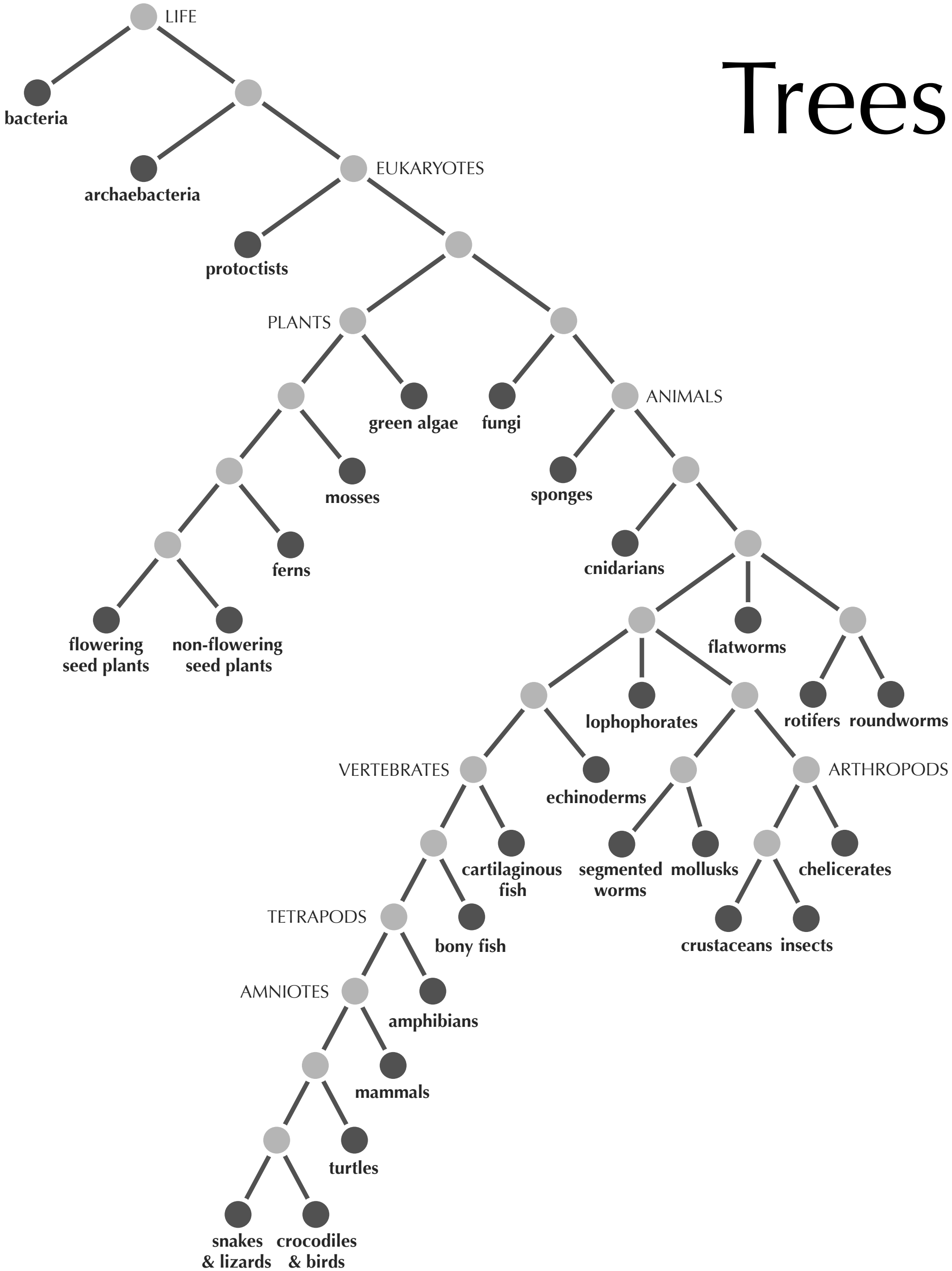
Trees



Tree: Connected graph containing no cycles.

Leaves (degree = 1): present-day species

Trees



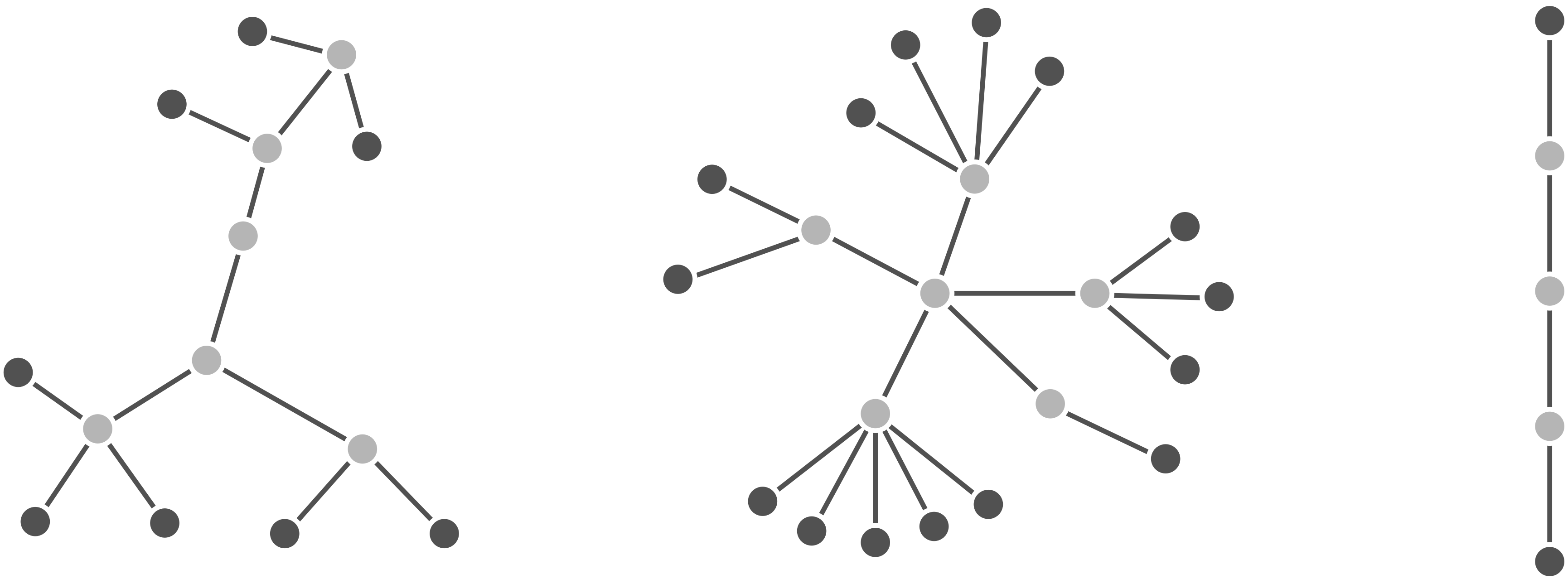
Tree: Connected graph containing no cycles.

Leaves (degree = 1): present-day species

Internal nodes (degree ≥ 2): ancestral species

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

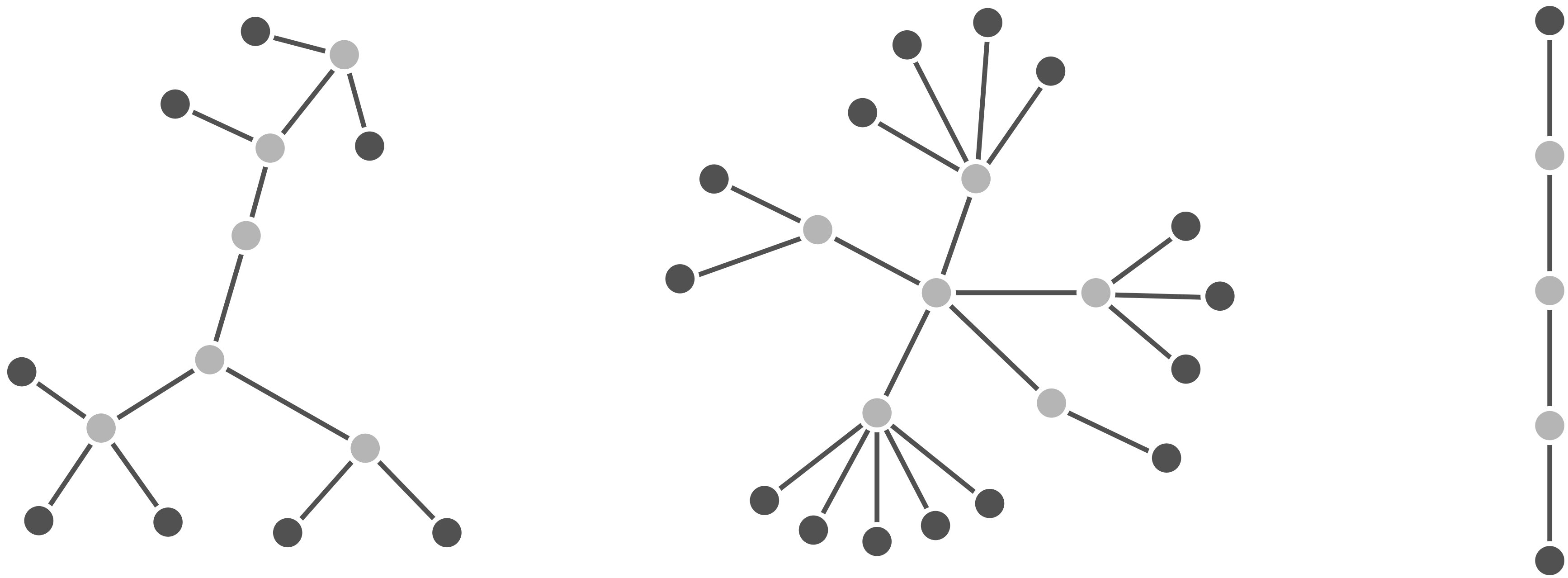
Trees



Note: We proved in a previous lecture that every tree with n nodes has exactly $n - 1$ edges.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

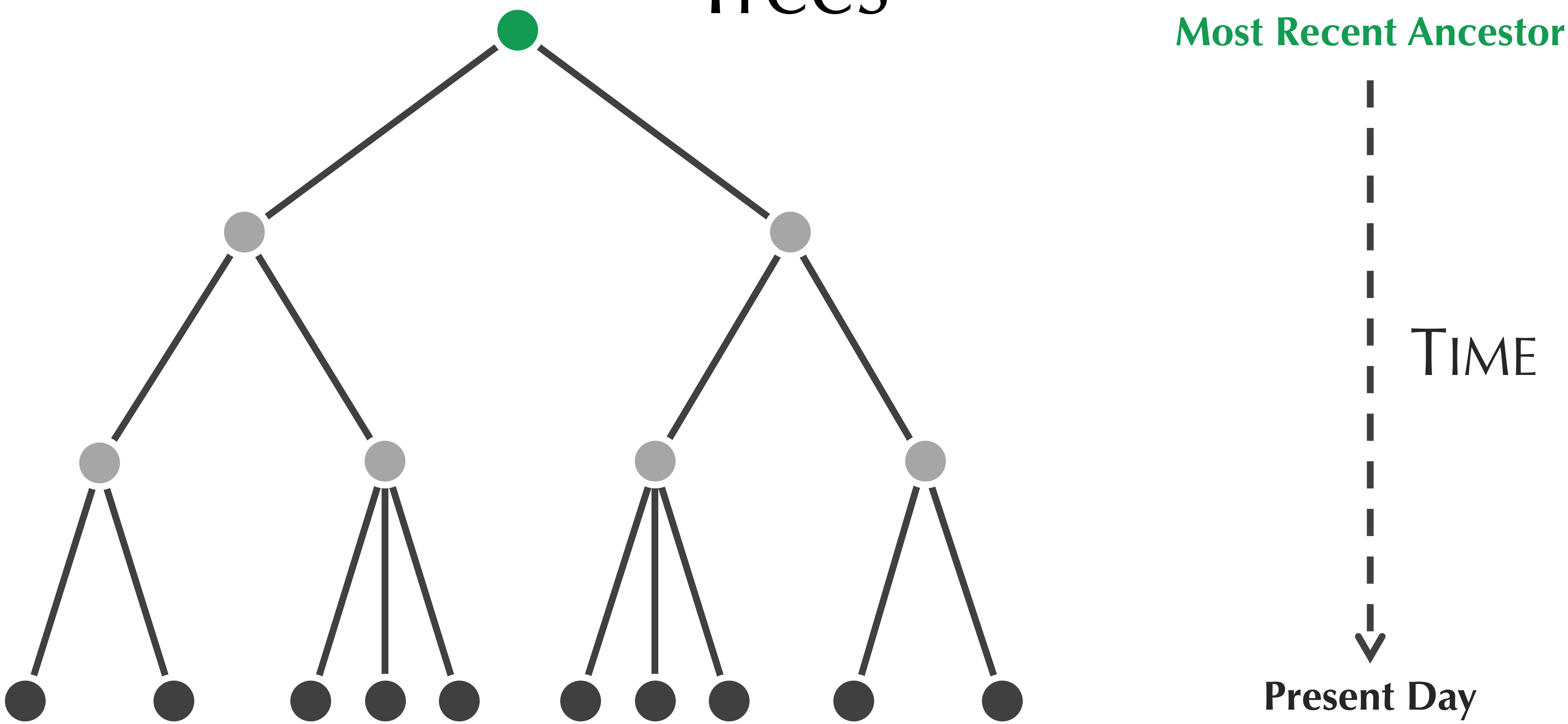
Trees



Exercise: Prove that there is a unique path connecting any two nodes in a tree.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Trees



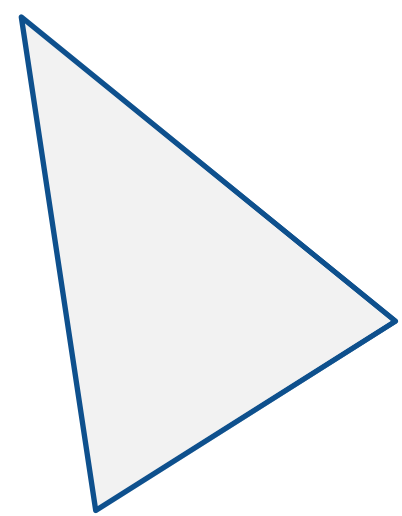
Rooted tree: one node is designated as the **root** (most recent common ancestor).

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Definition of a Distance Matrix

Distance matrix: A matrix D representing distances between pairs of n organisms that satisfies three properties:

1. **Symmetry:** $D_{i,j} = D_{i,j}$ for all pairs i, j
2. **Non-negativity:** $D_{i,j} >= 0$ for all pairs i, j
3. **Triangle inequality:** For all i, j, and k, $D_{i,j} + D_{j,k} >= D_{i,k}$.



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Multiple Alignment Defines a Simple Distance Matrix

SPECIES	ALIGNMENT
Chimp	`ACGTAGGCCT`
Human	`ATGTAAGACT`
Seal	`TCGAGAGCAC`
Whale	`TCGAAAGCAT`

A Multiple Alignment Defines a Simple Distance Matrix

$D_{i,j}$ = number of differing symbols between i-th and j-th rows of a multiple alignment.

SPECIES	ALIGNMENT	DISTANCE MATRIX			
		Chimp	**Human**	**Seal**	**Whale**
Chimp	`ACGTAGGCCT`	0	3	6	4
Human	`ATGTAAGACT`	3	0	7	5
Seal	`TCGAGAGCAC`	6	7	0	2
Whale	`TCGAAAGCAT`	4	5	2	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Multiple Alignment Defines a Simple Distance Matrix

$D_{i,j}$ = number of differing symbols between i-th and j-th rows of a multiple alignment.

Species	Alignment	Chimp	Human	Seal	Whale
		Distance Matrix			
Chimp	A**C**GTA**G**G**C**CT	0	**3**	6	4
Human	A**T**GTA**A**G**A**CT	**3**	0	7	5
Seal	TCGAGAGCAC	6	7	0	2
Whale	TCGAAAGCAT	4	5	2	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Multiple Alignment Defines a Simple Distance Matrix

Exercise: Prove that for any multiple sequence alignment, this way of defining *D* produces a distance matrix.

Species	Alignment	Chimp	Human	Seal	Whale
Chimp	ACGTAGGCCT	0	**3**	6	4
Human	ATGTAAGACT	**3**	0	7	5
Seal	TCGAGAGCAC	6	7	0	2
Whale	TCGAAAGCAT	4	5	2	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Distance-Based Phylogeny

Distance-Based Phylogeny Problem.
- **Input:** A distance matrix.
- **Output:** The unrooted tree "fitting" this distance matrix.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Distance-Based Phylogeny

Distance-Based Phylogeny Problem.
- **Input:** A distance matrix.
- **Output:** The unrooted tree "*fitting*" this distance matrix.

Of course, we are getting a bit ahead of ourselves – we should define what we mean by "fitting"!

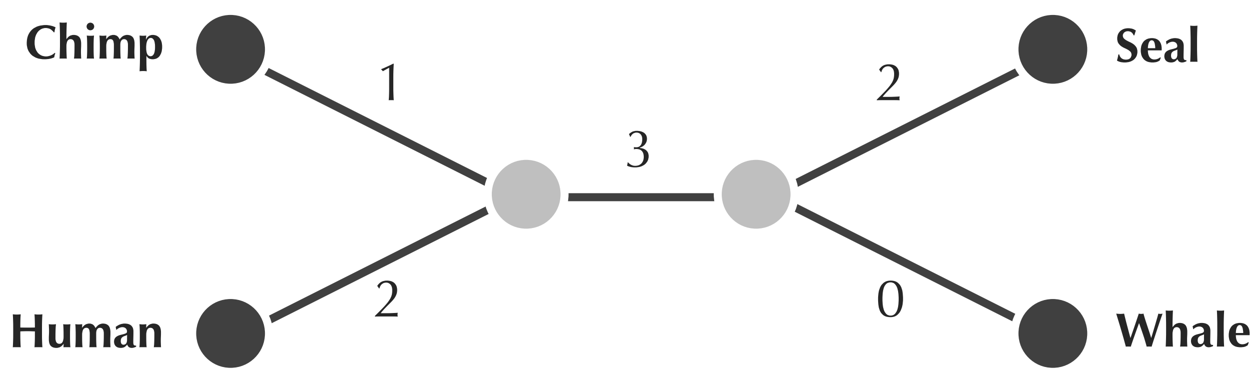
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

"Fitting" a Tree to a Matrix

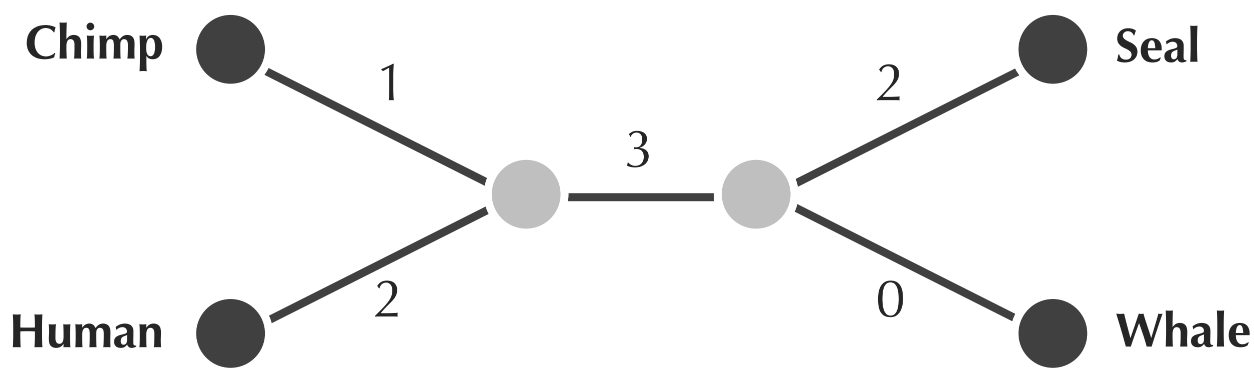
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

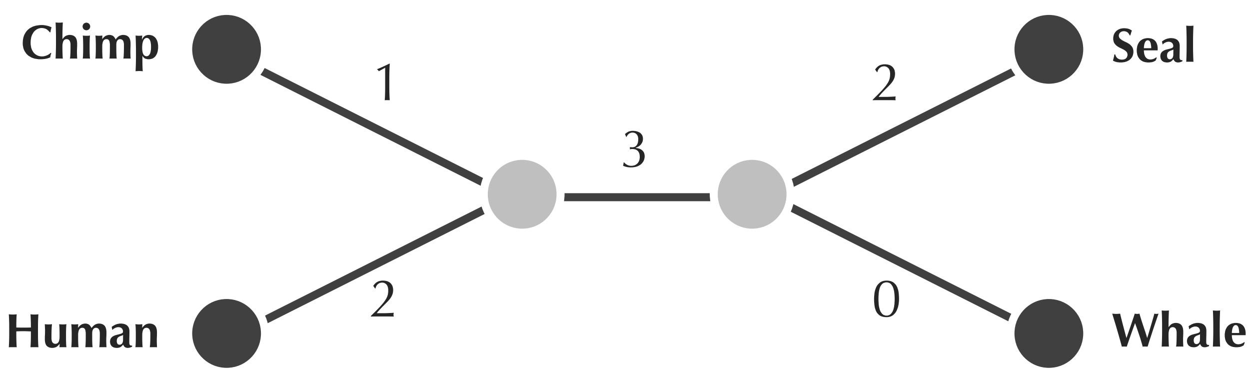


$d_{i,j}(T)$ = distance between nodes i and j in tree T, computed by summing edge weights from i to j.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

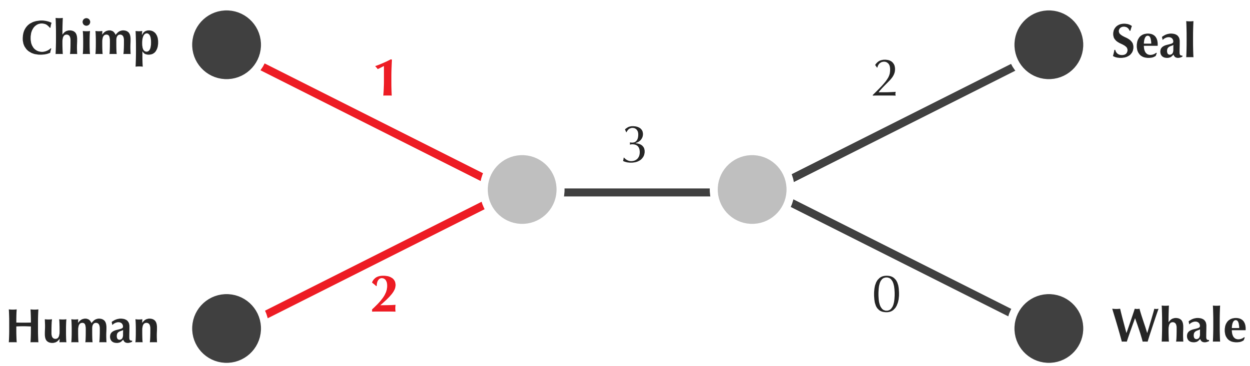


We say that T **fits** matrix D if for every pair i and j, $d_{i,j}(T) = D_{i,j}$.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

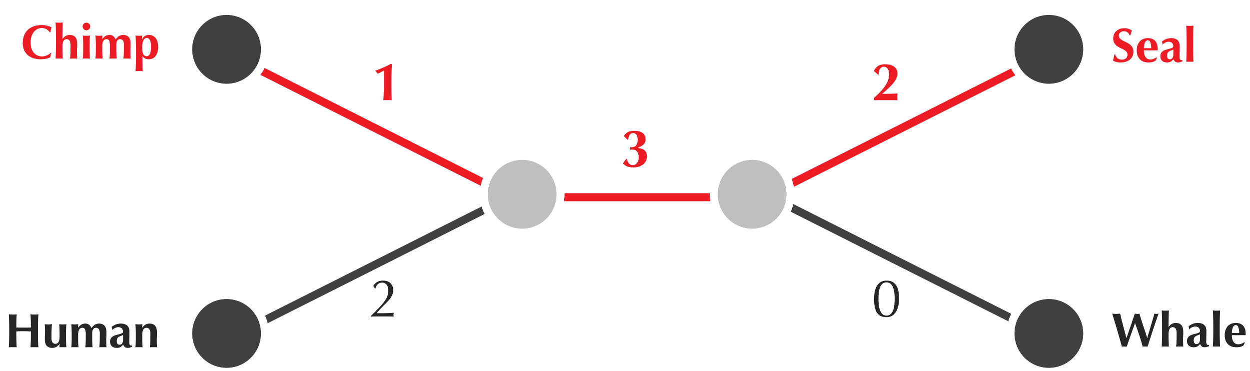
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

“Fitting” a Tree to a Matrix

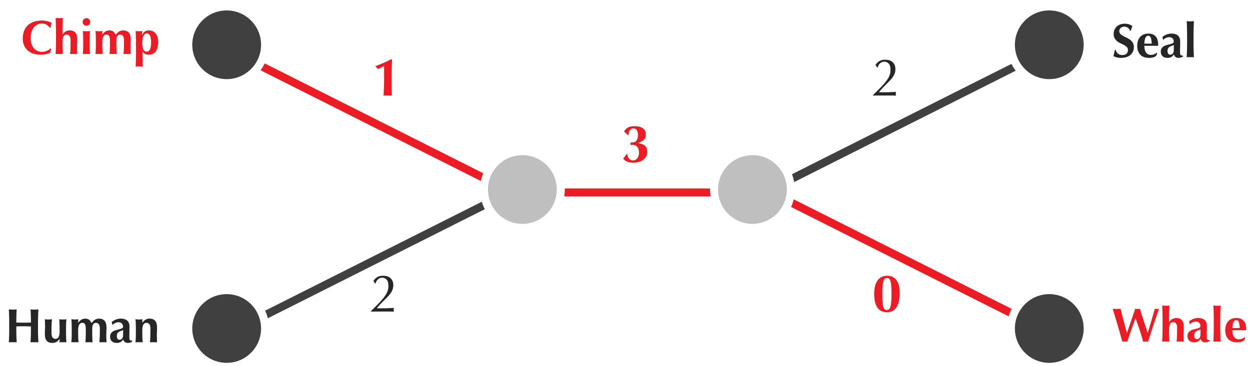
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

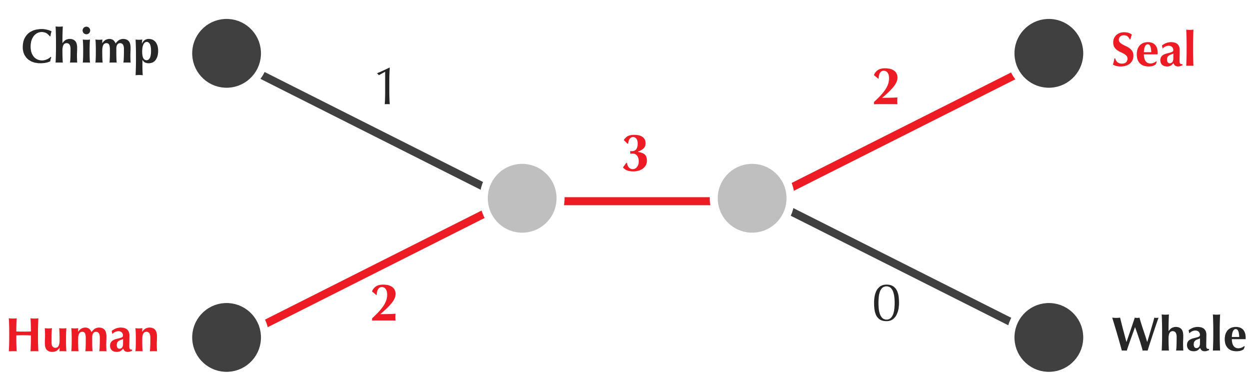
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

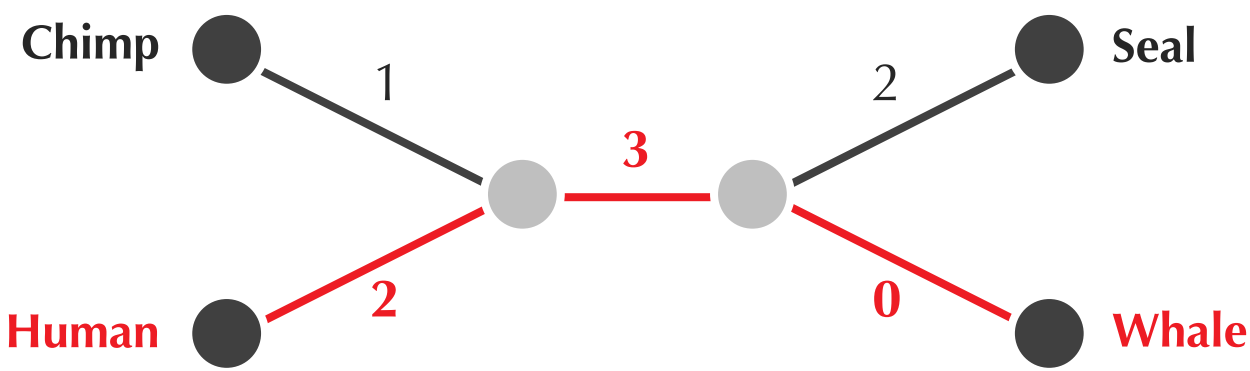
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

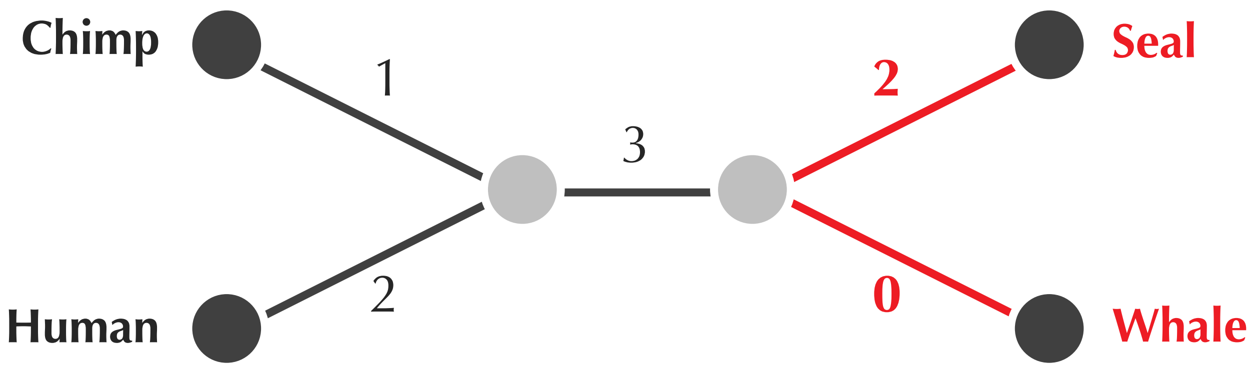
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	**5**
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

"Fitting" a Tree to a Matrix

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	**2**
Whale	4	5	2	0



Return to Distance-Based Phylogeny

Exercise: Find a tree fitting the following matrix.

	v_1	v_2	v_3	v_4
v_1	0	3	4	3
v_2	3	0	4	5
v_3	4	4	0	2
v_4	3	5	2	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Sometimes, No Tree Fits a Matrix

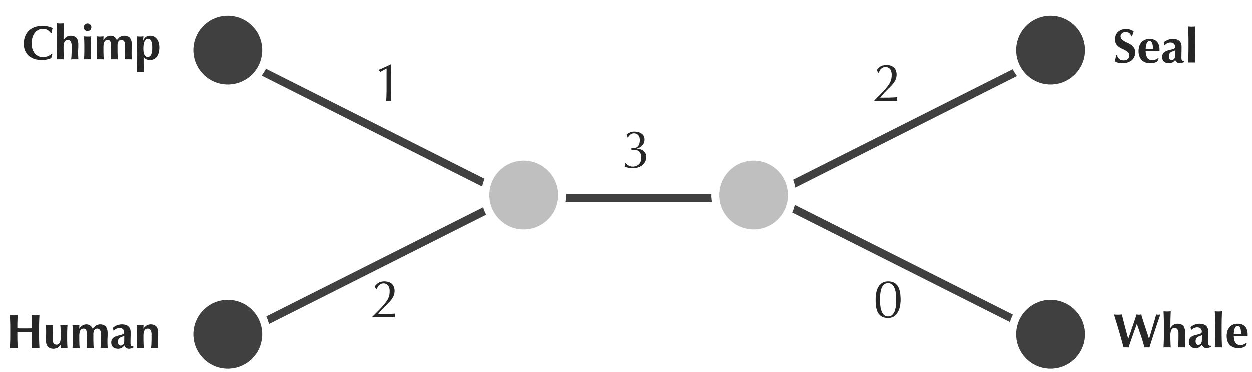
Exercise: Find a tree fitting the following matrix.

	v_1	v_2	v_3	v_4
v_1	0	3	4	3
v_2	3	0	4	5
v_3	4	4	0	2
v_4	3	5	2	0

Additive matrix: distance matrix such that there exists an unrooted tree fitting it.

Sometimes, **More Than One** Tree Fits a Matrix

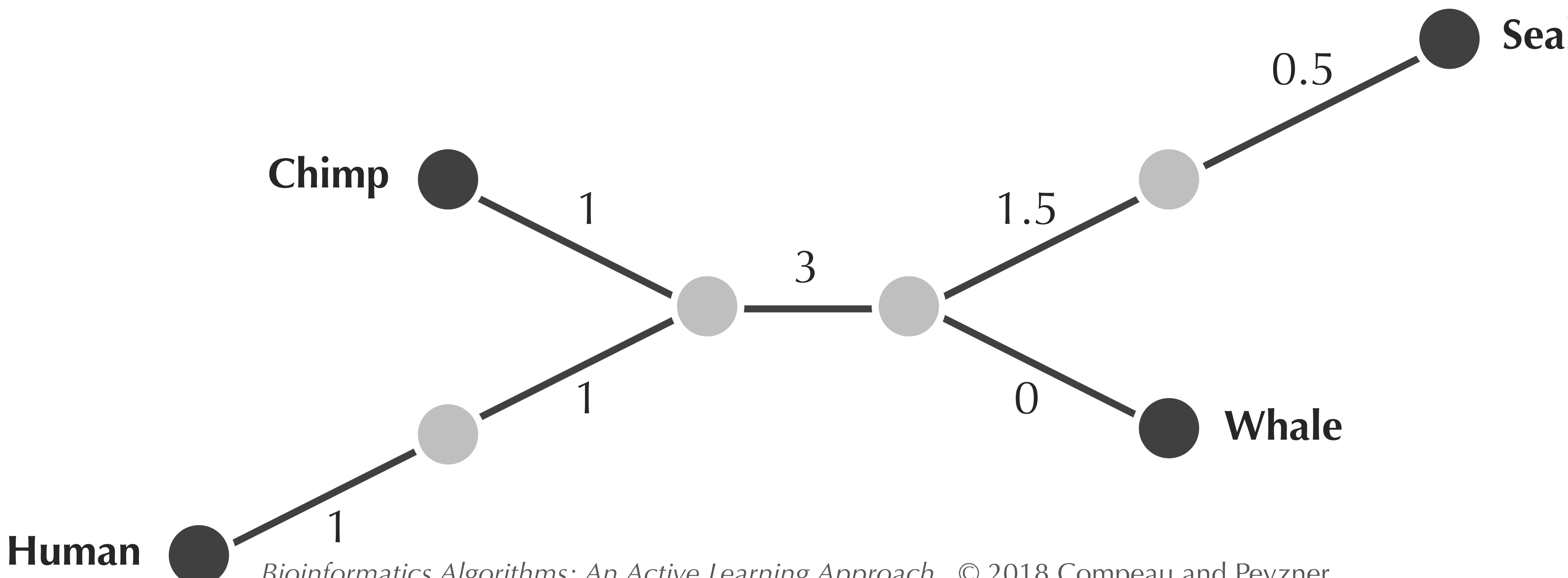
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

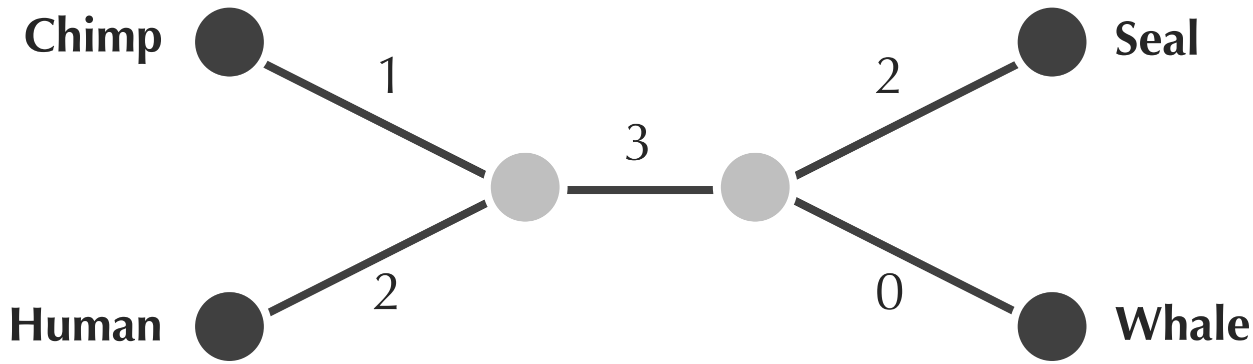
Sometimes, More Than One Tree Fits a Matrix

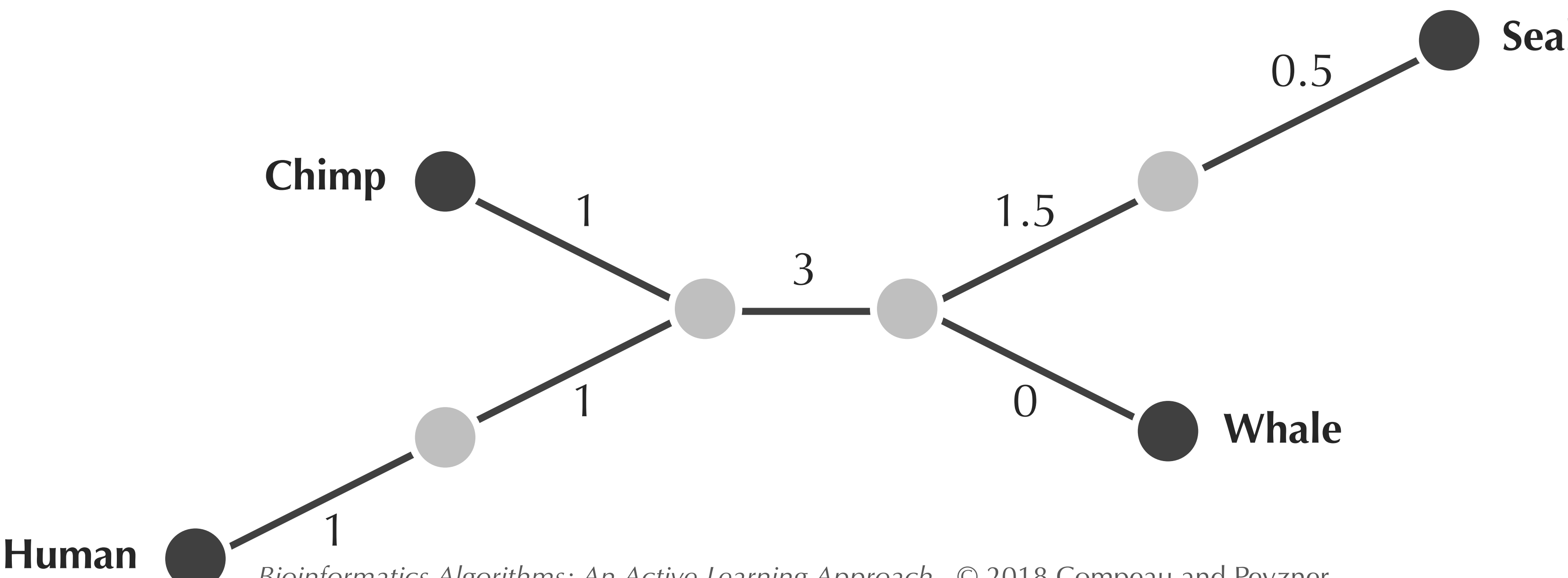
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

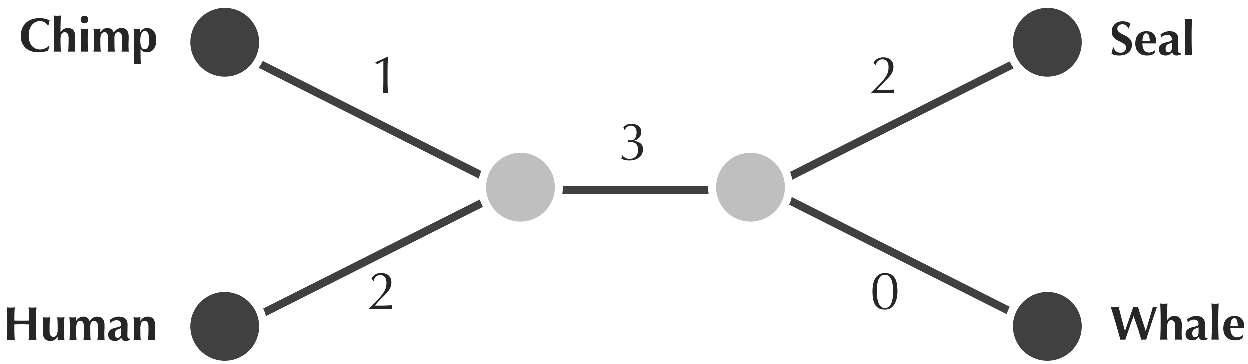
Which Tree is Better?

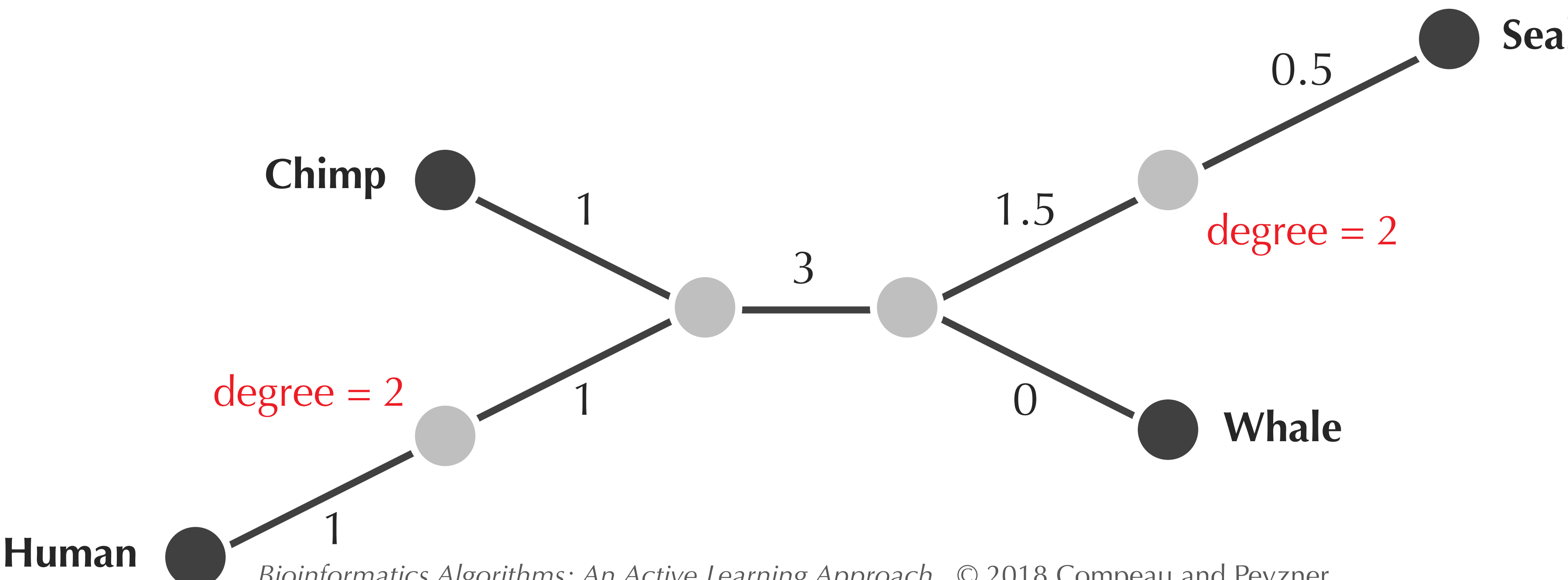




Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

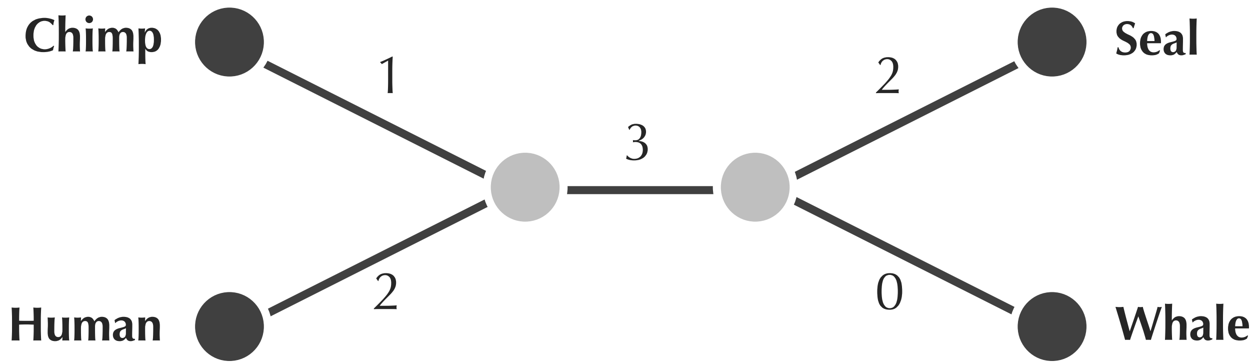
Which Tree is Better?





Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

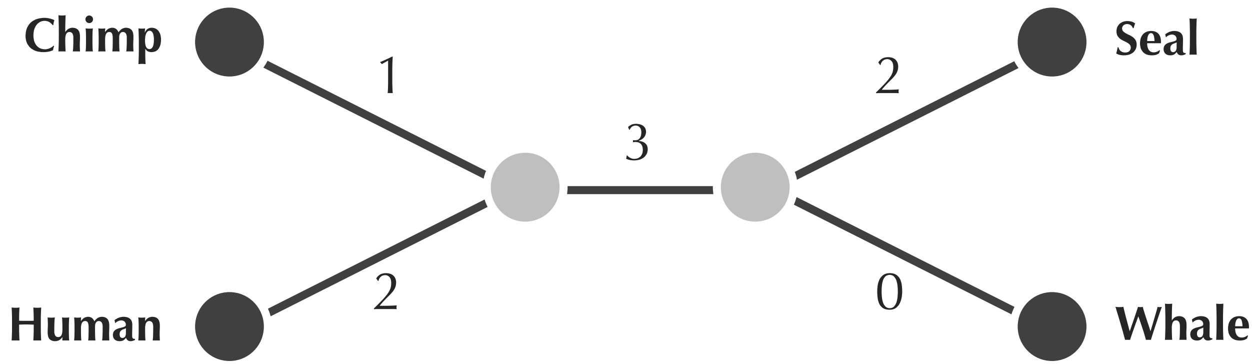
Which Tree is Better?



Simple tree: tree with no nodes of degree 2.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Which Tree is Better?



Simple tree: tree with no nodes of degree 2.

Theorem: There is a unique *simple* tree fitting an *additive* matrix.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

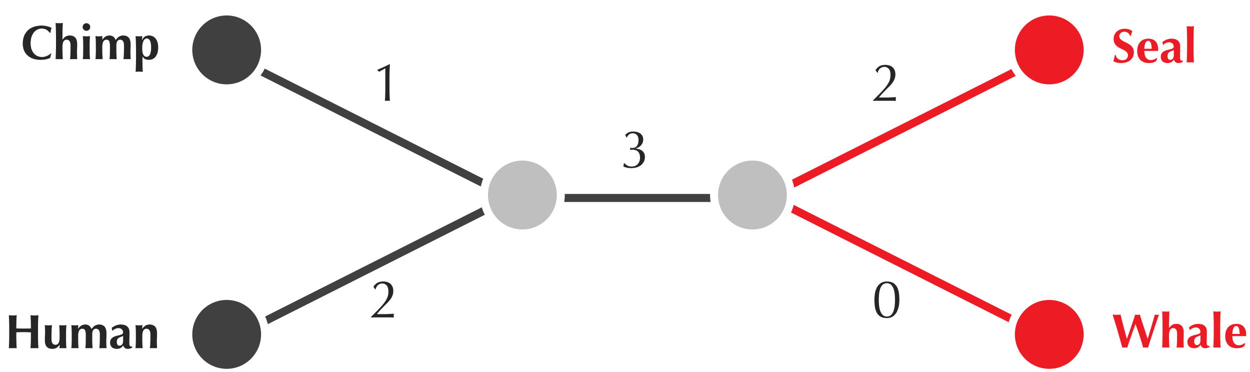
Reformulating Distance-Based Phylogeny

Distance-Based Phylogeny Problem: *Construct an evolutionary tree from a distance matrix.*

- **Input:** A distance matrix.
- **Output:** The simple tree fitting this distance matrix (if this matrix is additive).

An Idea for Distance-Based Phylogeny

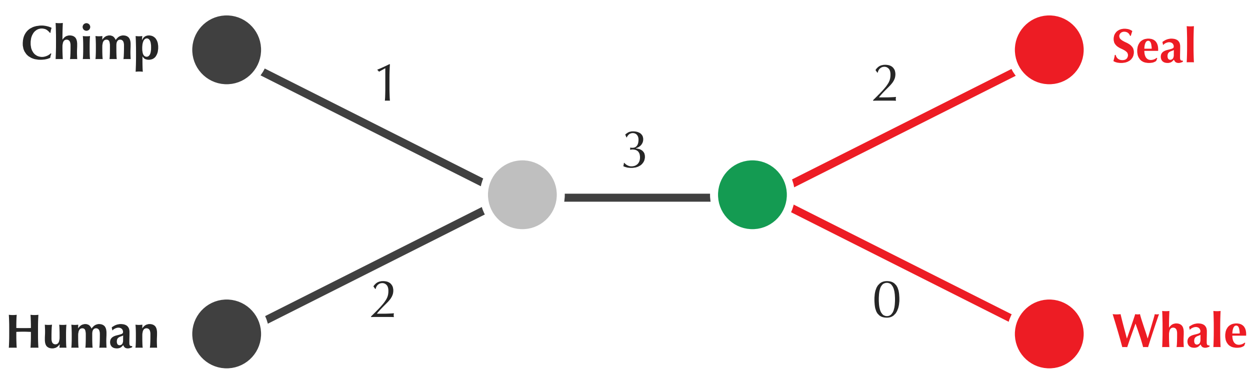
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	**2**
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

Seal and whale are **neighbors** (meaning they share the same **parent**).

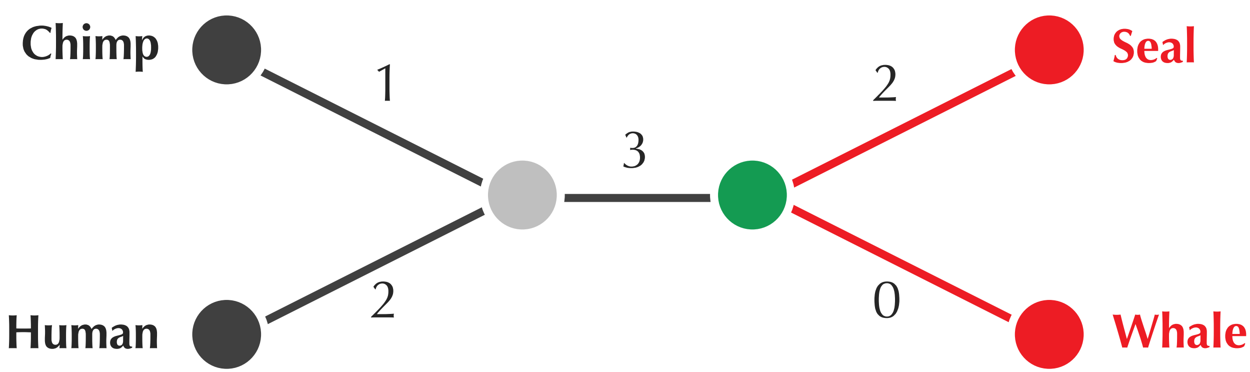


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

Seal and whale are **neighbors** (meaning they share the same **parent**).

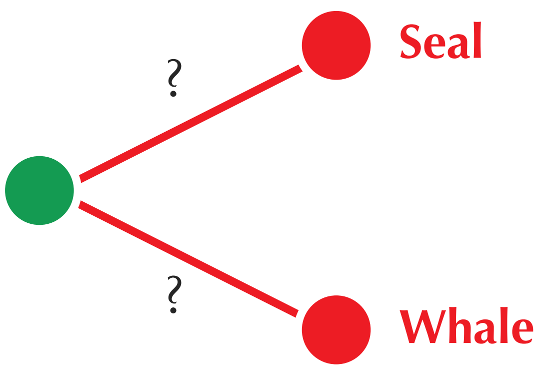
Theorem: Every simple tree with at least three leaves has at least one pair of neighboring leaves.



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	**2**
Whale	4	5	2	0

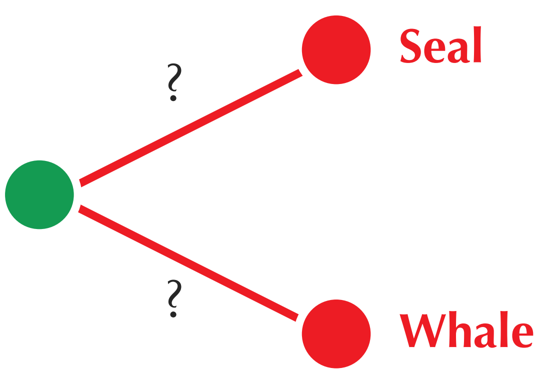


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

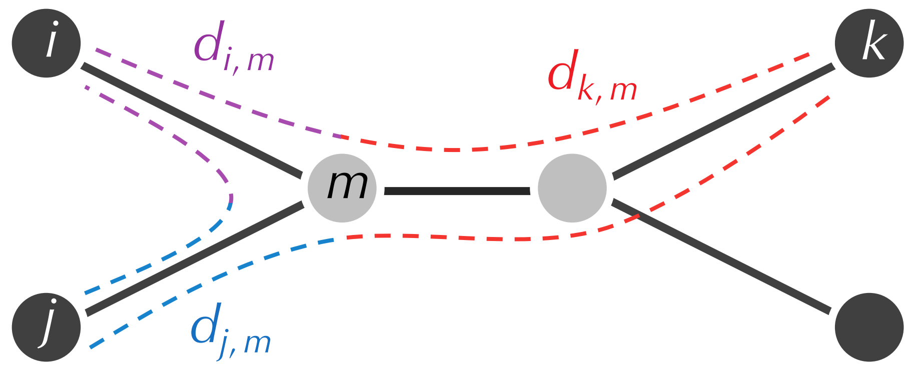
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	**2**
Whale	4	5	2	0

Key Point: How do we compute the unknown distances?

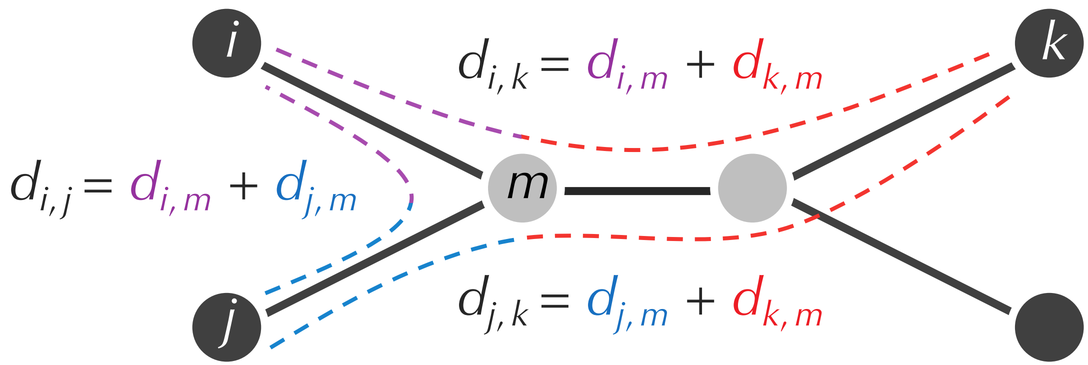


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

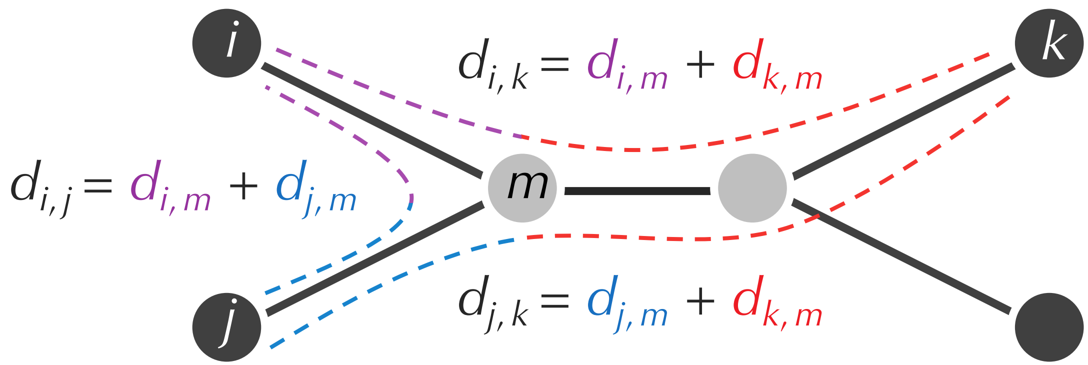
Toward a Recursive Algorithm



Toward a Recursive Algorithm



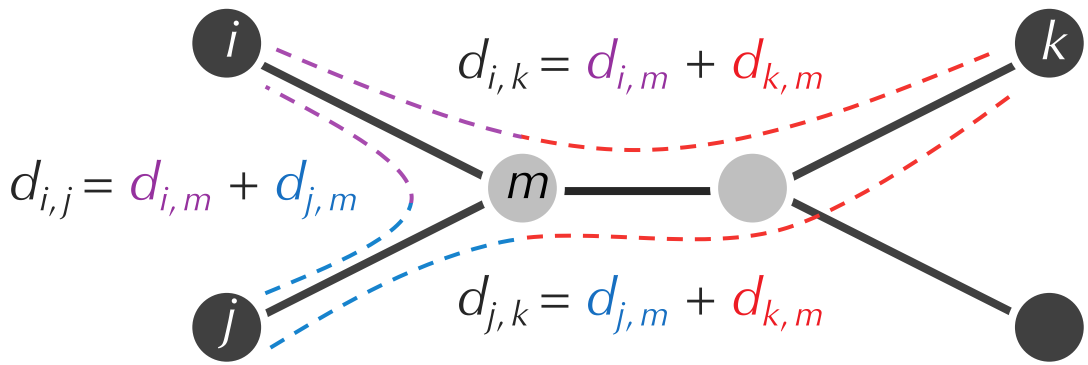
Toward a Recursive Algorithm



$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] \;/\; 2$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Recursive Algorithm

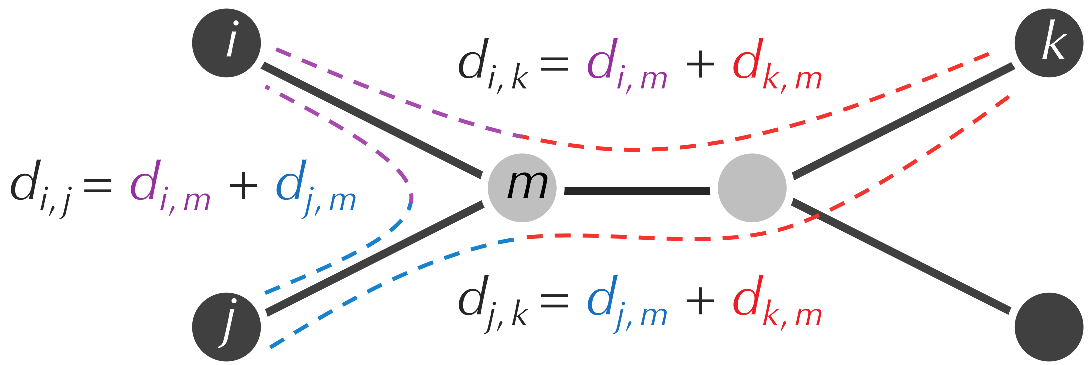


$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Recursive Algorithm



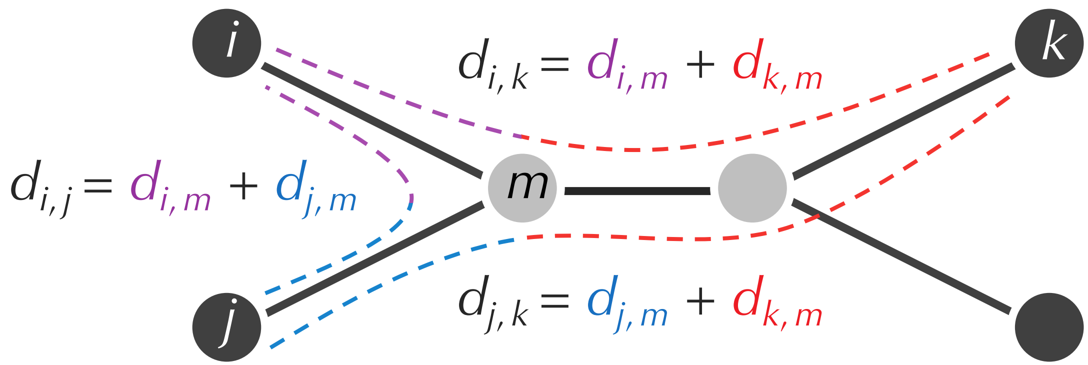
$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Recursive Algorithm



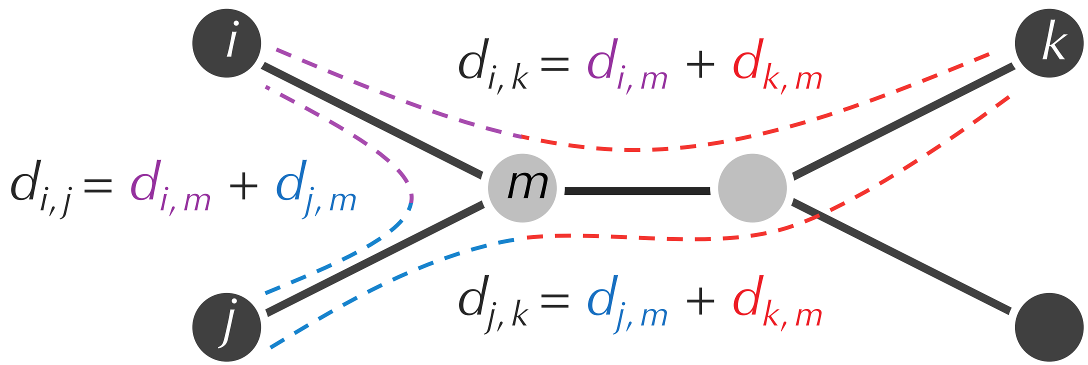
$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$\therefore \quad d_{i,m} = D_{i,k} - (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

Toward a Recursive Algorithm



$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

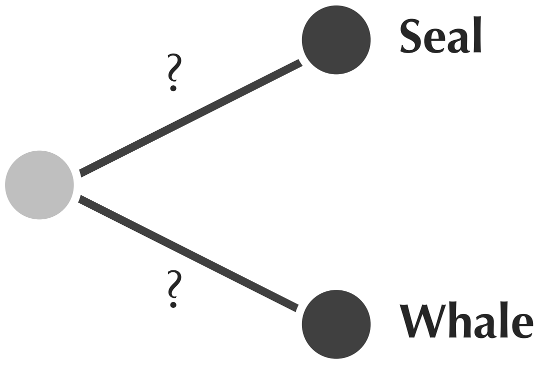
$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$\therefore \quad d_{i,m} = D_{i,k} - (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$d_{i,m} = (D_{i,k} + D_{i,j} - D_{j,k}) / 2$$

An Idea for Distance-Based Phylogeny

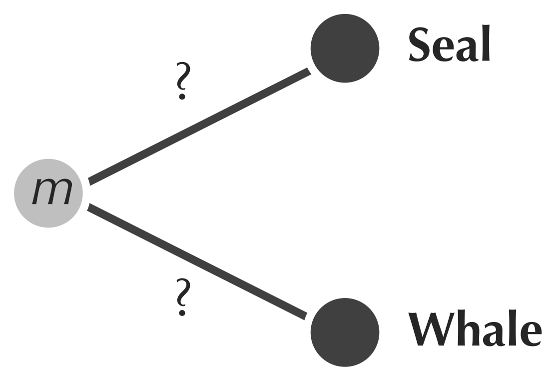
	Chimp	**Human**	**Seal**	**Whale**
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{i,m} = (D_{i,k} + D_{i,j} - D_{j,k}) / 2$$

An Idea for Distance-Based Phylogeny

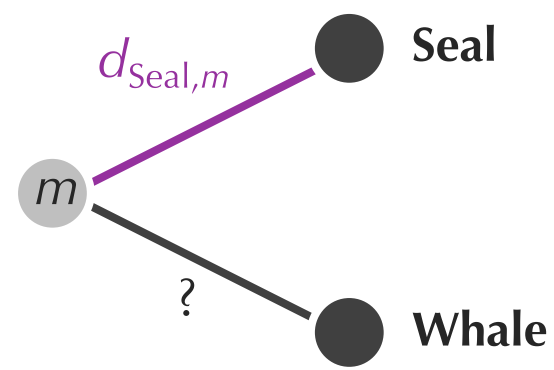
	Chimp	**Human**	**Seal**	**Whale**
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{i,m} = (D_{i,k} + D_{i,j} - D_{j,k}) / 2$$

An Idea for Distance-Based Phylogeny

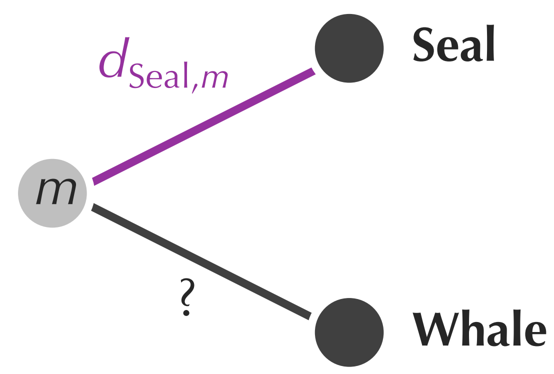
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (D_{\text{Seal},k} + D_{\text{Seal},j} - D_{j,k}) \,/\, 2$$

An Idea for Distance-Based Phylogeny

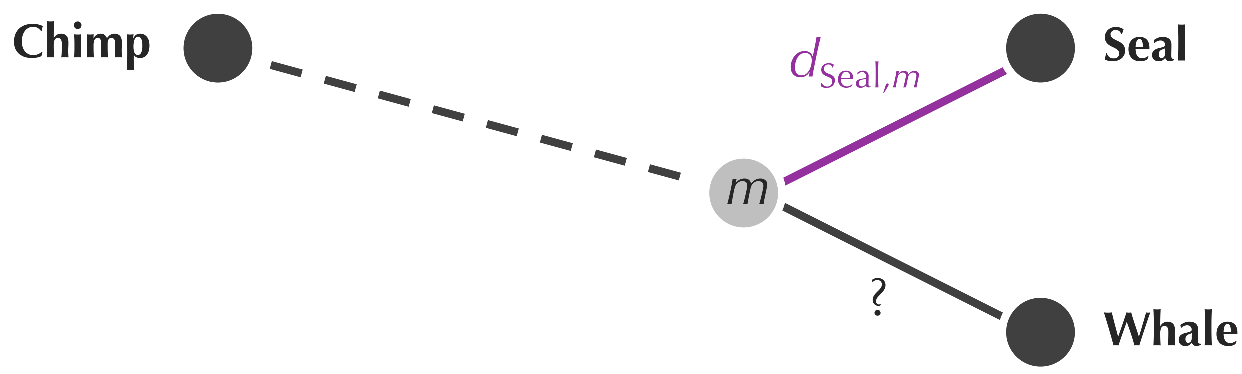
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (D_{\text{Seal},k} + D_{\text{Seal,Whale}} - D_{\text{Whale},k}) \,/\, 2$$

An Idea for Distance-Based Phylogeny

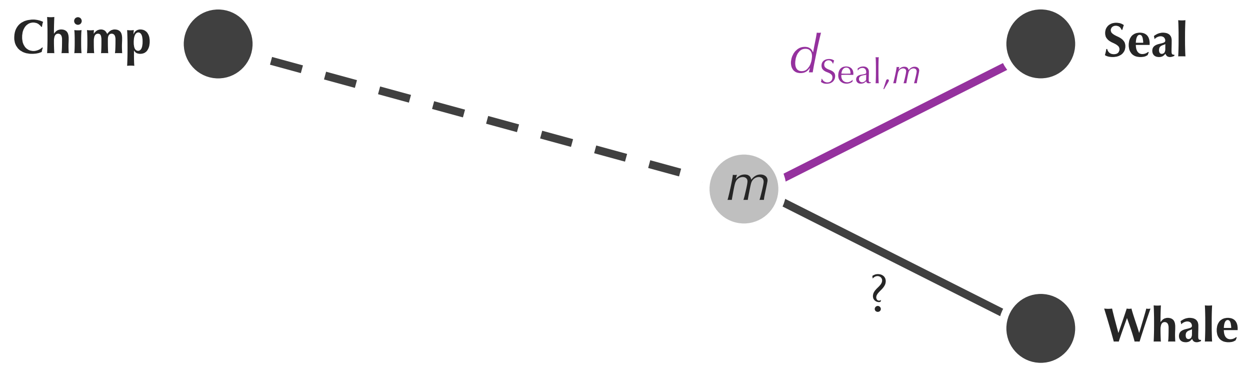
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (D_{\text{Seal,Chimp}} + D_{\text{Seal,Whale}} - D_{\text{Whale,Chimp}}) \,/\, 2$$

An Idea for Distance-Based Phylogeny

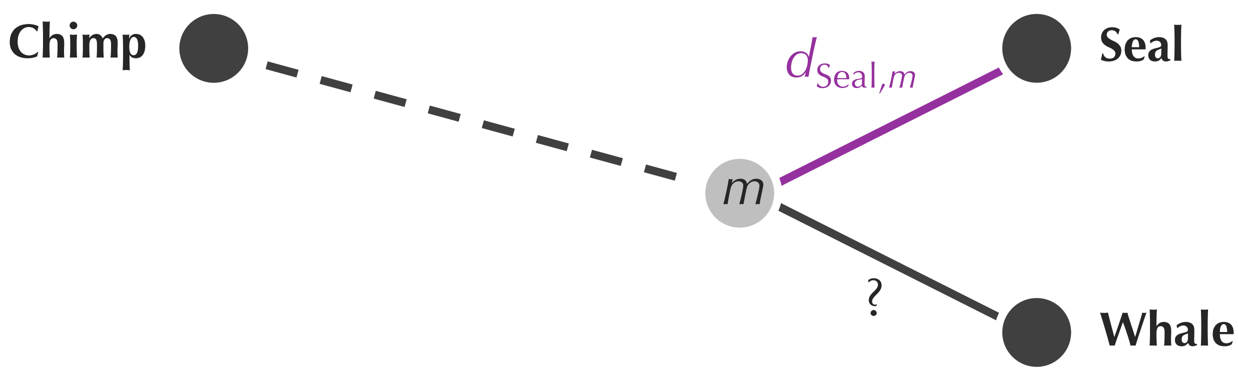
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (\quad 6 \quad + D_{\text{Seal,Whale}} - D_{\text{Whale,Chimp}}) / 2$$

An Idea for Distance-Based Phylogeny

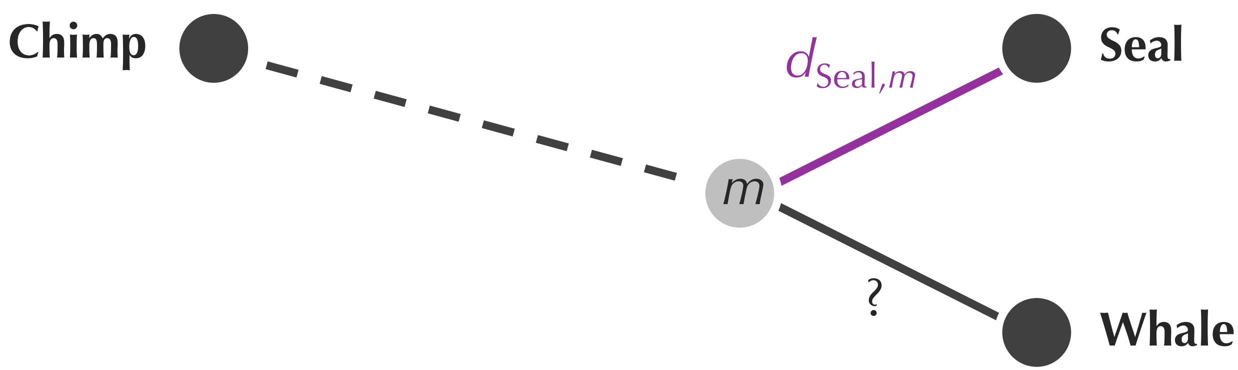
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (\quad 6 \quad + \quad 2 \quad - D_{\text{Whale,Chimp}})\,/\,2$$

An Idea for Distance-Based Phylogeny

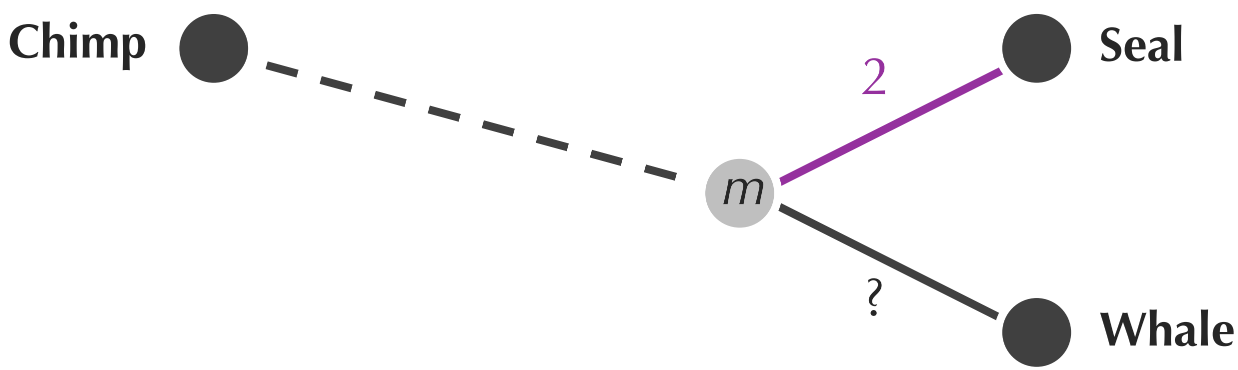
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$$d_{\text{Seal},m} = (\quad 6 \quad + \quad 2 \quad - \quad 4 \quad)\,/\,2$$

An Idea for Distance-Based Phylogeny

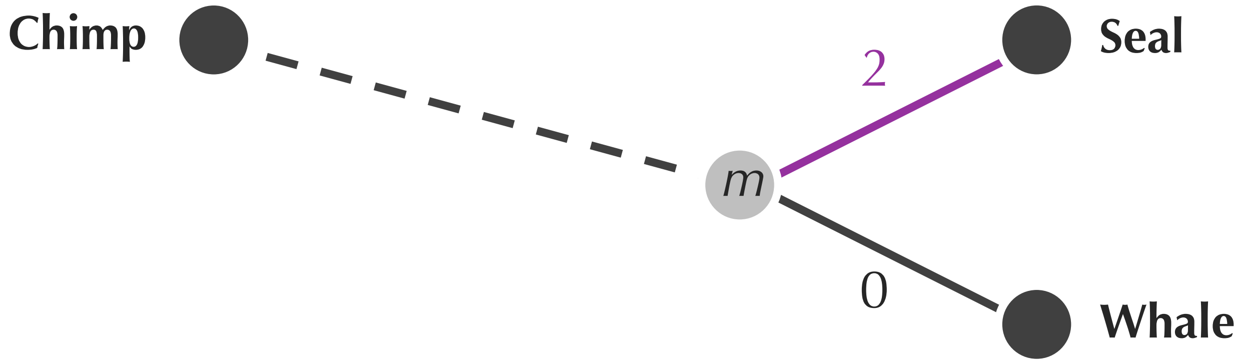
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$d_{\mathrm{Seal},m} = 2$

An Idea for Distance-Based Phylogeny

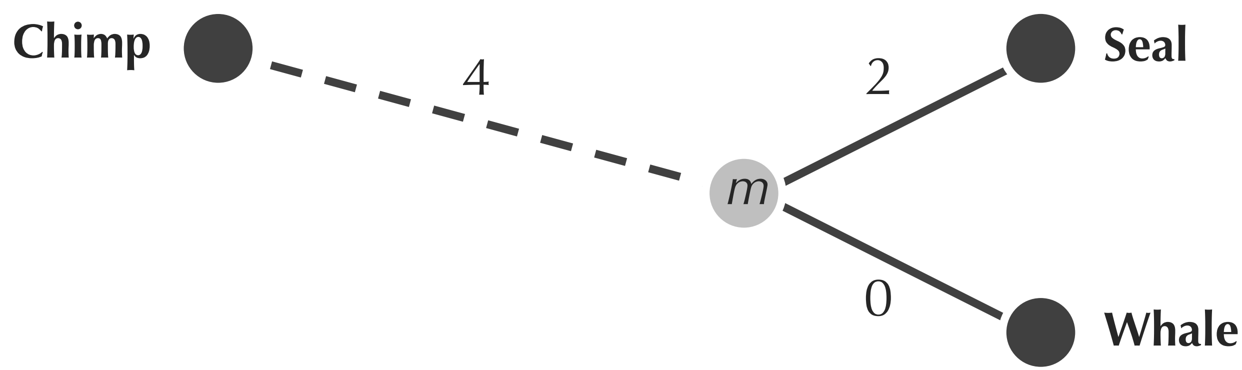
	Chimp	**Human**	**Seal**	**Whale**
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



$d_{\text{Seal},m} = 2$

An Idea for Distance-Based Phylogeny

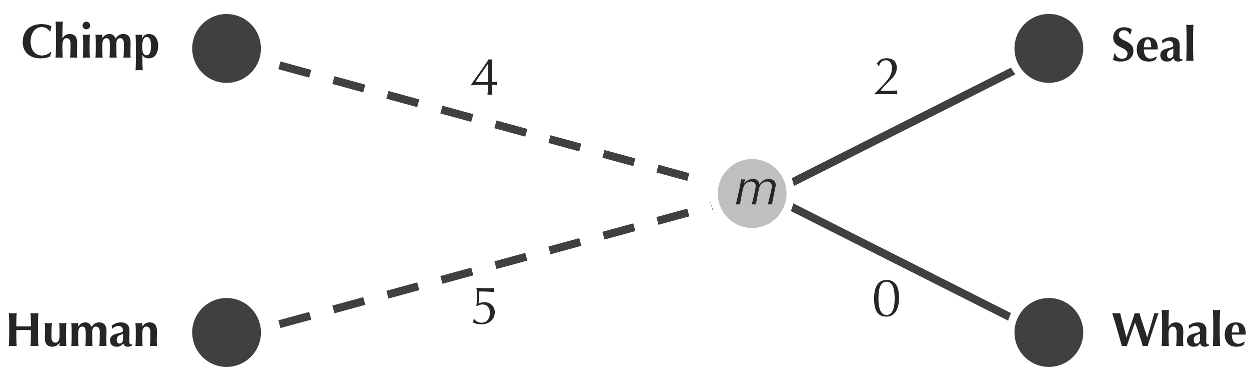
	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

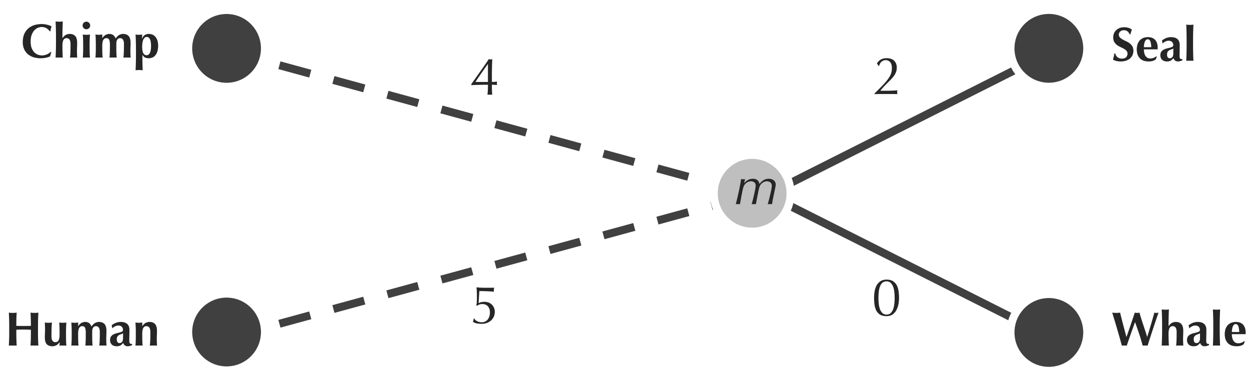
	Chimp	Human	Seal	Whale	*m*
Chimp	0	3	6	4	4
Human	3	0	7	5	5
Seal	6	7	0	2	2
Whale	4	5	2	0	0
m	4	5	2	0	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

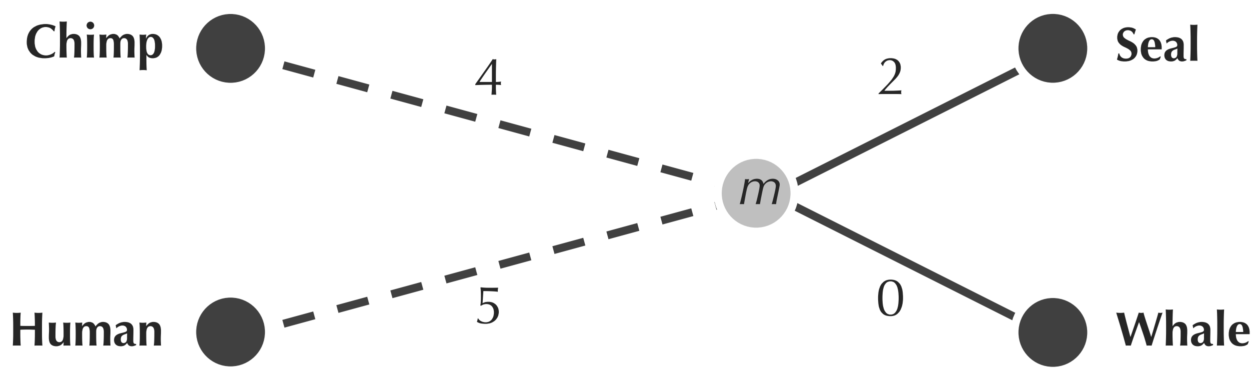
	Chimp	Human	Seal	Whale	*m*
Chimp	0	3	6	4	4
Human	3	0	7	5	5
Seal	6	7	0	2	2
Whale	4	5	2	0	0
m	4	5	2	0	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

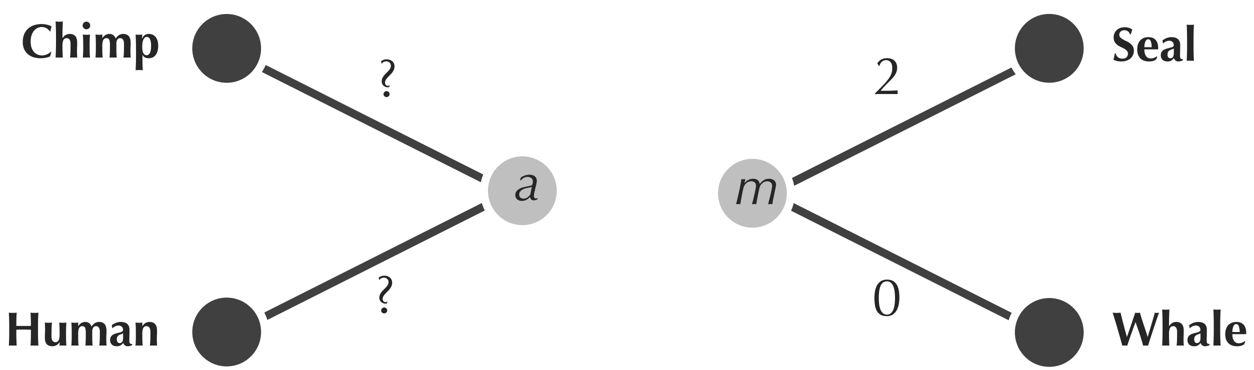
	Chimp	Human	*m*
Chimp	0	3	4
Human	3	0	5
m	4	5	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

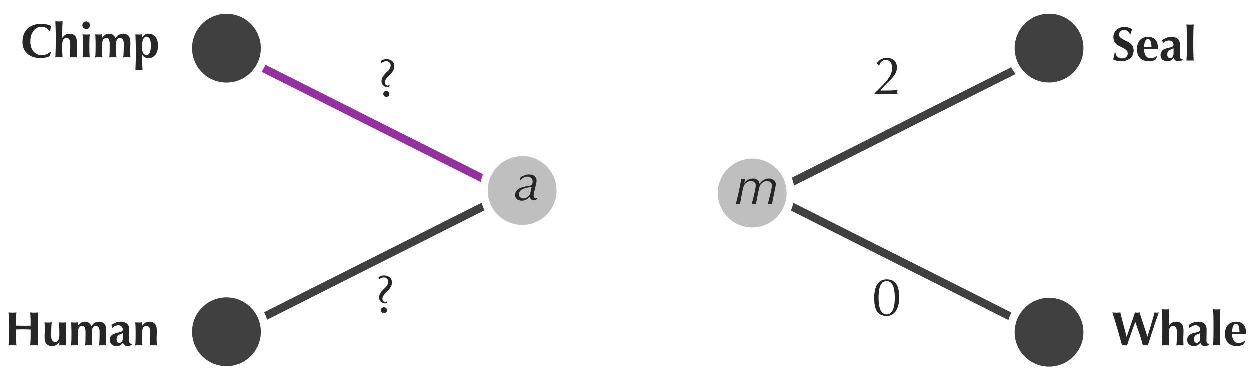
An Idea for Distance-Based Phylogeny

	Chimp	Human	*m*
Chimp	0	**3**	4
Human	**3**	0	5
m	4	5	0



An Idea for Distance-Based Phylogeny

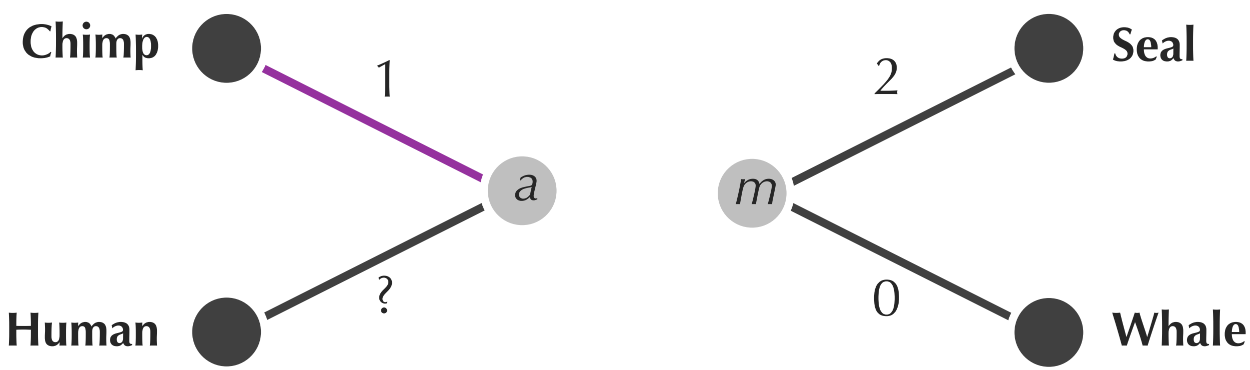
	Chimp	Human	*m*
Chimp	0	3	4
Human	3	0	5
m	4	5	0



$$d_{\text{Chimp},a} = (D_{\text{Chimp},m} + D_{\text{Chimp},\text{Human}} - D_{\text{Human},m}) / 2$$

An Idea for Distance-Based Phylogeny

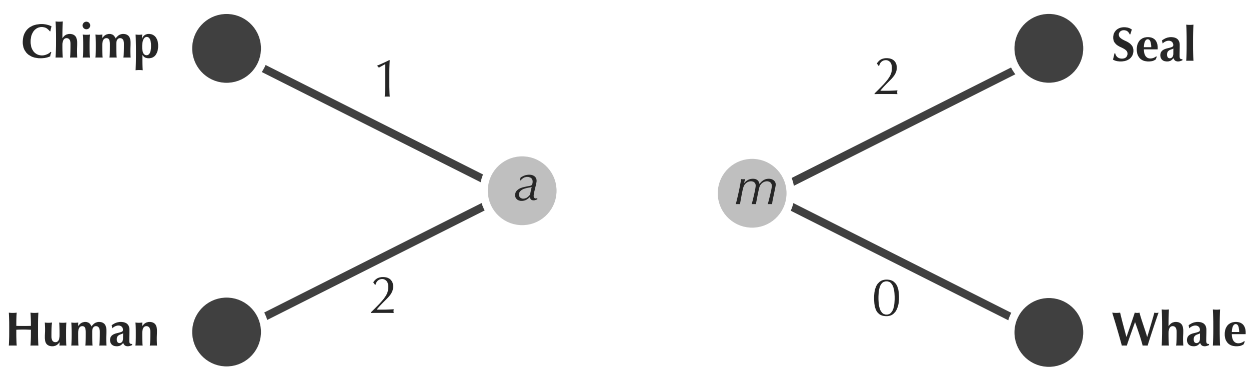
	Chimp	**Human**	***m***
Chimp	0	3	4
Human	3	0	5
m	4	5	0



$d_{\text{Chimp},a} = 1$

An Idea for Distance-Based Phylogeny

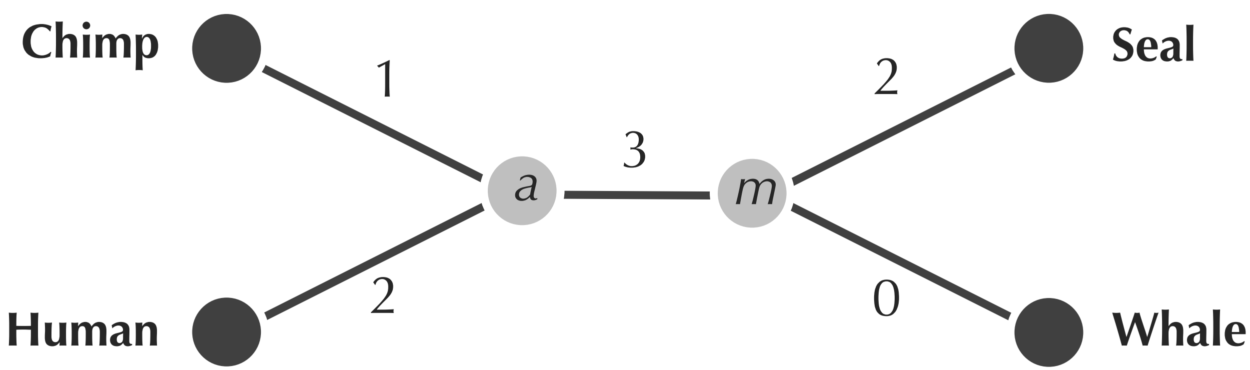
	Chimp	**Human**	***m***
Chimp	0	3	4
Human	3	0	5
m	4	5	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

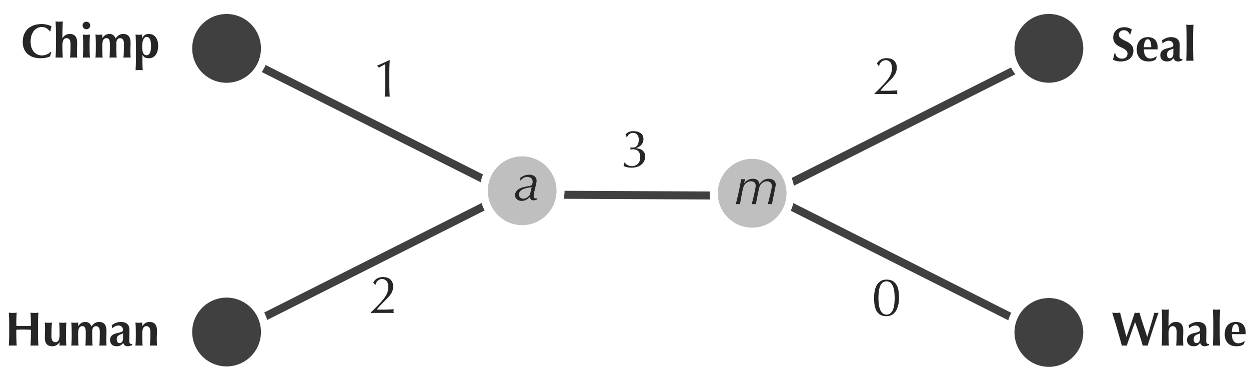
	Chimp	Human	*m*
Chimp	0	3	4
Human	3	0	5
m	4	5	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Idea for Distance-Based Phylogeny

	0	**1**	**2**	**3**
0	0	13	21	22
1	13	0	12	13
2	21	12	0	13
3	22	13	13	0

Exercise: Apply this recursive approach to this distance matrix.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

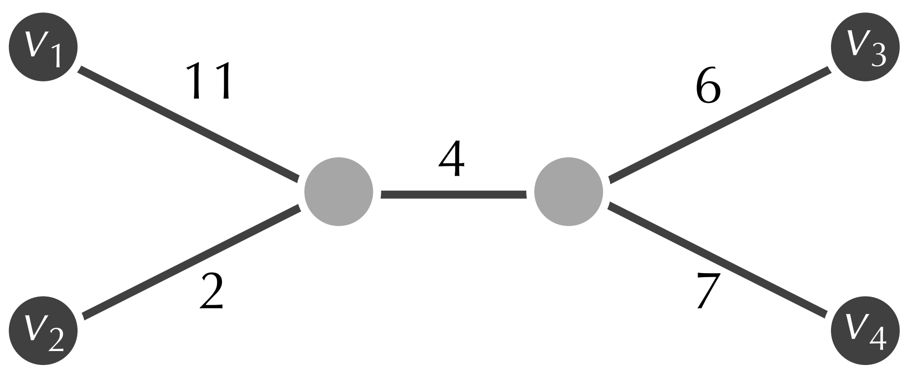
What Was Wrong With Our Algorithm?

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

What Was Wrong With Our Algorithm?

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

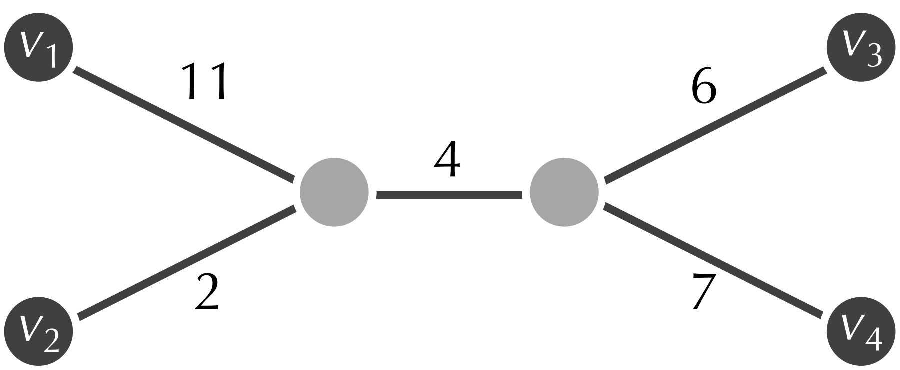


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

What Was Wrong With Our Algorithm?

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	**12**	13
v_3	21	12	0	13
v_4	22	13	13	0

minimum element is $D_{2,3}$

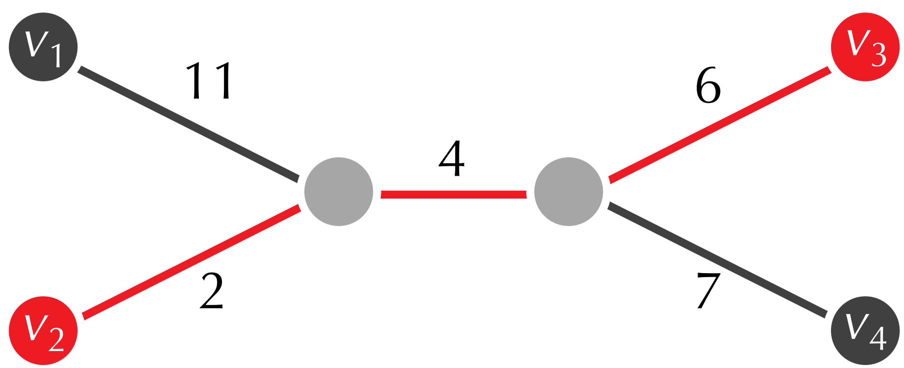


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

What Was Wrong With Our Algorithm?

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	**12**	13
v_3	21	12	0	13
v_4	22	13	13	0

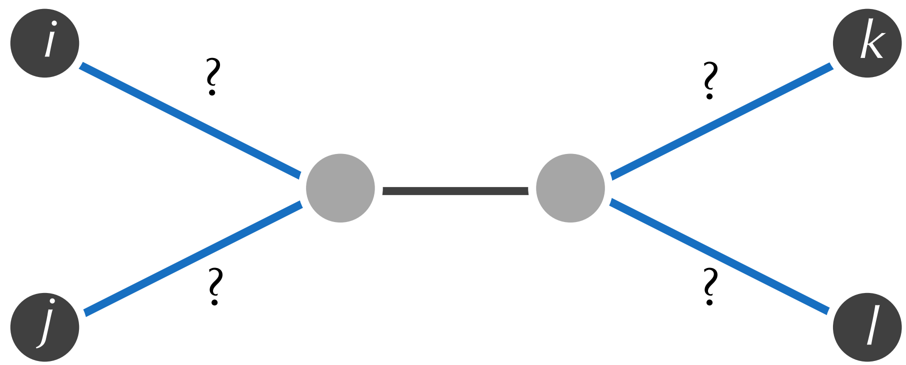
minimum element is $D_{2,3}$



v_2 and v_3 are **not** neighbors!

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

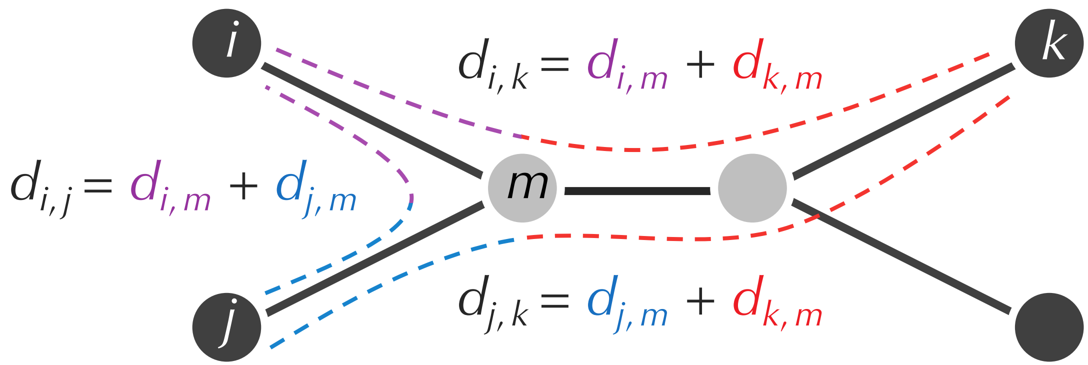
From Neighbors to Limbs



Rather than trying to infer **neighbors**, let's instead try to compute the length of **limbs**, the edges attached to leaves.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

From Neighbors to Limbs



$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

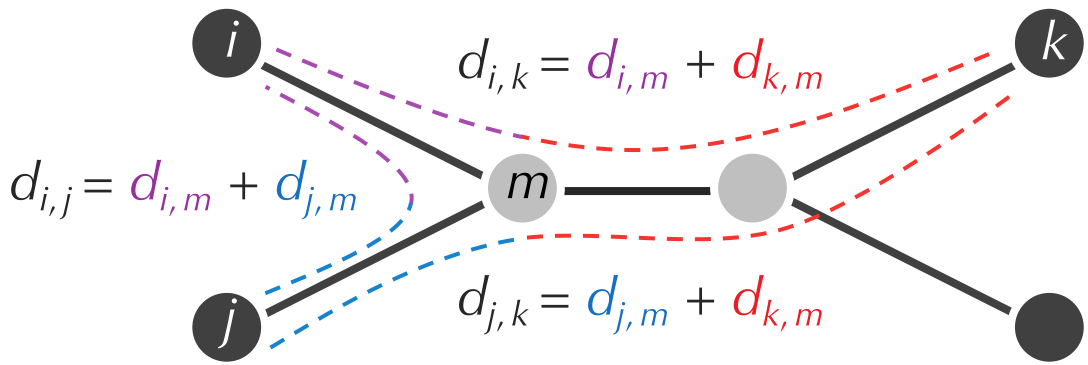
$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$\therefore \quad d_{i,m} = D_{i,k} - (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$d_{i,m} = (D_{i,k} + D_{i,j} - D_{j,k}) / 2$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

From Neighbors to Limbs



$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$\therefore \quad d_{i,m} = D_{i,k} - (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

$$d_{i,m} = (D_{i,k} + D_{i,j} - D_{j,k}) / 2$$

Assumes that *i* and *j* are *neighbors*...

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(i) is equal to the minimum value of $(D_{i,k} + D_{i,j} - D_{j,k})/2$ over all leaves j and k.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	**Human**	**Seal**	**Whale**
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

$(D_{\text{chimp, human}} + D_{\text{chimp, seal}} - D_{\text{human, seal}})\ /\ 2 \quad = (3 + 6 - 7)\ /\ 2 = 1$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

$$(D_{\text{chimp, human}} + D_{\text{chimp, seal}} - D_{\text{human, seal}}) / 2 = (3 + 6 - 7) / 2 = 1$$

$$(D_{\text{chimp, human}} + D_{\text{chimp, whale}} - D_{\text{human, whale}}) / 2 = (3 + 4 - 5) / 2 = 1$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

$(D_{\text{chimp, human}} + D_{\text{chimp, seal}} - D_{\text{human, seal}}) / 2 = (3 + 6 - 7) / 2 = 1$

$(D_{\text{chimp, human}} + D_{\text{chimp, whale}} - D_{\text{human, whale}}) / 2 = (3 + 4 - 5) / 2 = 1$

$(D_{\text{chimp, whale}} + D_{\text{chimp, seal}} - D_{\text{whale, seal}}) / 2 = (6 + 4 - 2) / 2 = 4$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

$(D_{\text{chimp, human}} + D_{\text{chimp, seal}} - D_{\text{human, seal}}) / 2 = (3 + 6 - 7) / 2 = \mathbf{1}$

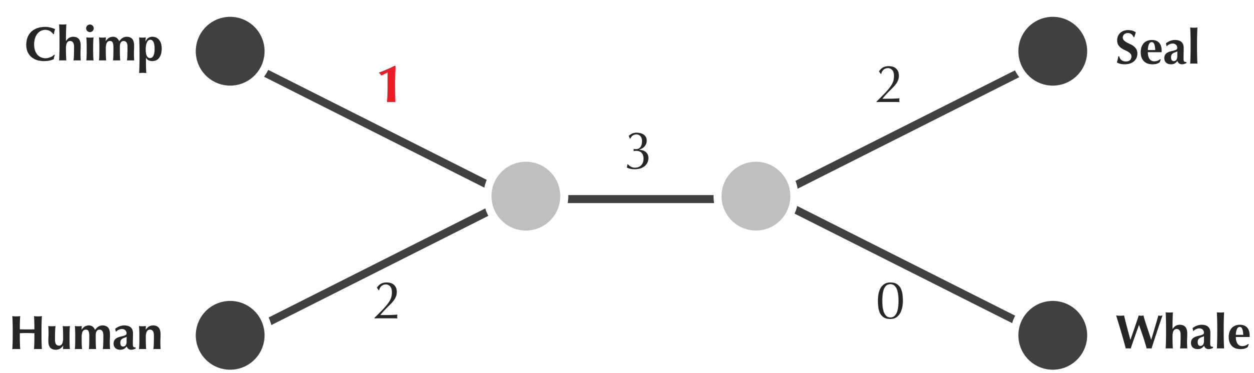
$(D_{\text{chimp, human}} + D_{\text{chimp, whale}} - D_{\text{human, whale}}) / 2 = (3 + 4 - 5) / 2 = \mathbf{1}$

$(D_{\text{chimp, whale}} + D_{\text{chimp, seal}} - D_{\text{whale, seal}}) / 2 = (6 + 4 - 2) / 2 = 4$

Computing Limb Lengths

Limb Length Theorem: *LimbLength*(chimp) is equal to the minimum value of $(D_{\text{chimp},k} + D_{\text{chimp},j} - D_{j,k})/2$ over all leaves j and k.

	Chimp	Human	Seal	Whale
Chimp	0	3	6	4
Human	3	0	7	5
Seal	6	7	0	2
Whale	4	5	2	0

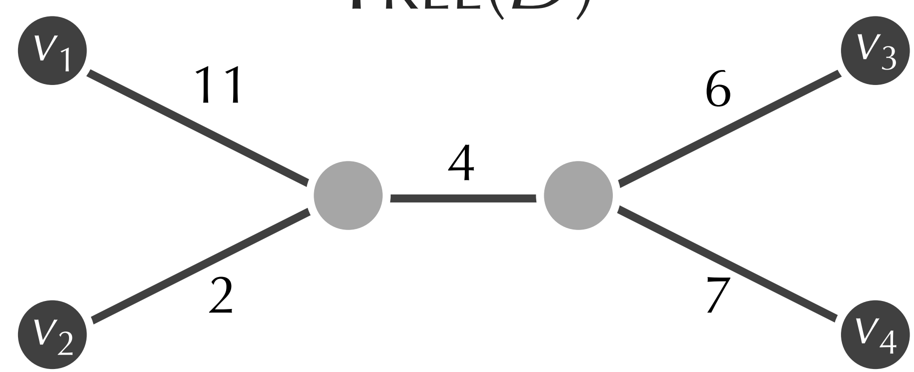


AdditivePhylogeny In Action

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

TREE(D)



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

AdditivePhylogeny In Action

$$D \quad \begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 13 & 21 & 22 \\ v_2 & 13 & 0 & 12 & 13 \\ v_3 & 21 & 12 & 0 & 13 \\ v_4 & 22 & 13 & 13 & 0 \end{array}$$

1. Pick an arbitrary leaf j (say, $j = v_2$).

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

AdditivePhylogeny In Action

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

$LimbLength(v_2) = 2$

2. Compute its limb length, $LimbLength(j)$.

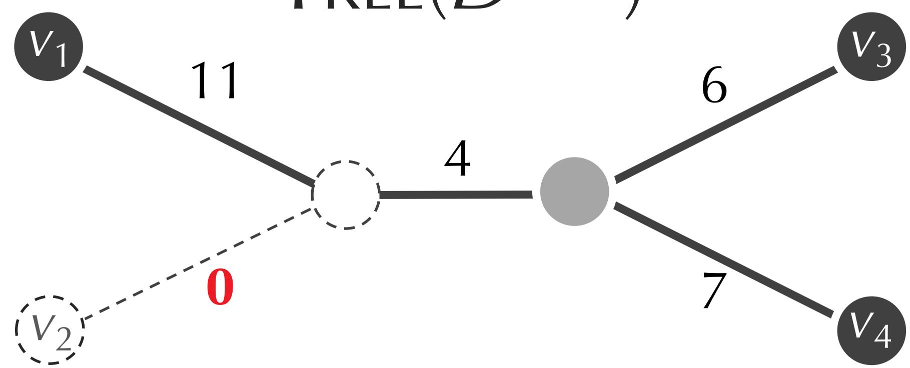
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

AdditivePhylogeny In Action

D^{bald}

	v_1	v_2	v_3	v_4
v_1	0	11	21	22
v_2	11	0	10	11
v_3	21	10	0	13
v_4	22	11	13	0

TREE(D^{bald})



3. Subtract *LimbLength*(*j*) from each row and column to produce D^{bald} in which *j* is a **bald limb** (length 0).

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

AdditivePhylogeny In Action

D^{trim}

	v_1	v_2	v_3	v_4
v_1	0	11	21	22
v_2	11	0	10	11
v_3	21	10	0	13
v_4	22	11	13	0

4. Remove the *j*-th row and column of the matrix to form the $(n - 1) \times (n - 1)$ matrix D^{trim}.

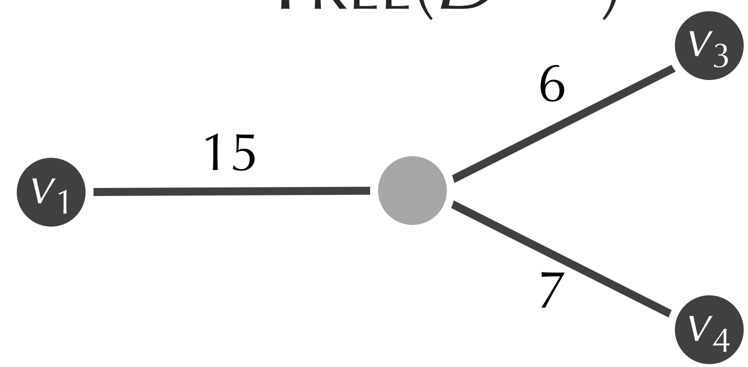
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

AdditivePhylogeny In Action

D^{trim}

	v_1	v_2	v_3	v_4
v_1	0	11	21	22
v_2	11	0	10	11
v_3	21	10	0	13
v_4	22	11	13	0

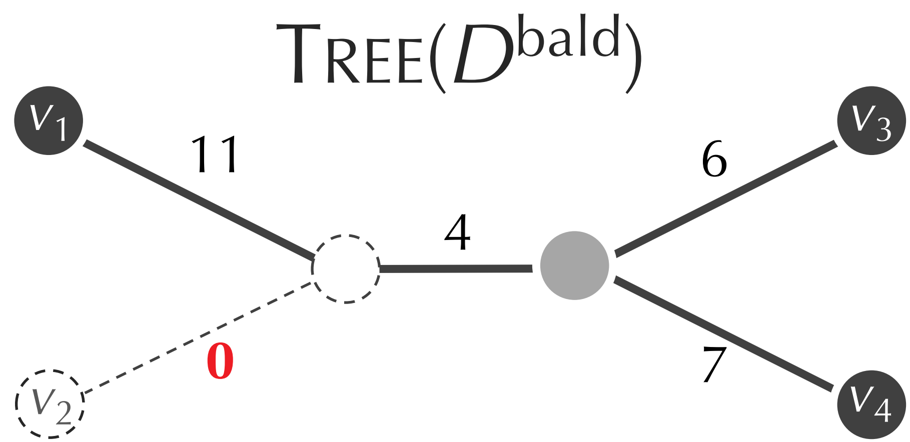
TREE(D^{trim})



5. Call AdditivePhylogeny recursively to construct *Tree*(D^{trim}).

AdditivePhylogeny In Action

$$D^{\text{bald}} \quad \begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 11 & 21 & 22 \\ v_2 & 11 & 0 & 10 & 11 \\ v_3 & 21 & 10 & 0 & 13 \\ v_4 & 22 & 11 & 13 & 0 \end{array}$$



6. Identify the point in *Tree*(D^{trim}) where leaf *j* should be attached.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

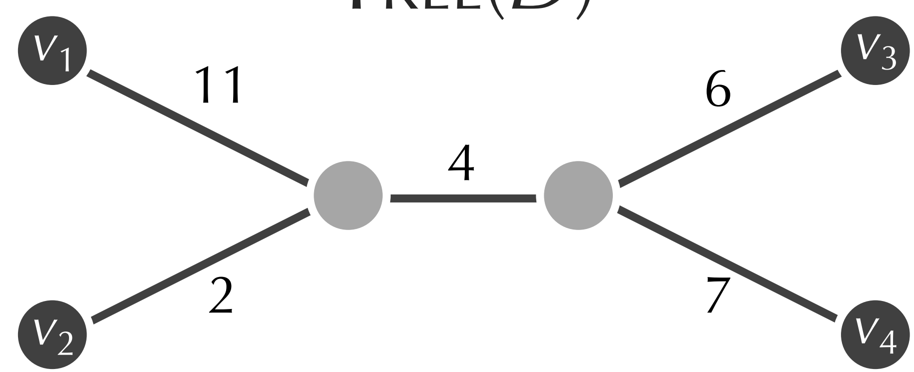
AdditivePhylogeny In Action

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

$LimbLength(v_2) = 2$

TREE(D)



7. Attach j by an edge of length $LimbLength(j)$ in order to form $Tree(D)$.

AdditivePhylogeny

AdditivePhylogeny(*D*):

1. Pick an arbitrary leaf *j*.
2. Compute its limb length, *LimbLength*(*j*).
3. Subtract *LimbLength*(*j*) from each row and column to produce D^{bald} in which *j* is a bald limb (length 0).
4. Remove the *j*-th row and column of the matrix to form the (n – 1) x (n – 1) matrix D^{trim}.
5. Recursively call **AdditivePhylogeny(**D^{trim}**)** to obtain *Tree*(D^{trim}).
6. **Identify the point in *Tree*(D^{trim}) where leaf *j* should be attached.**
7. Attach *j* by an edge of length *LimbLength*(*j*) in order to form *Tree*(*D*).

Attaching a leaf to the tree after the recursive step is the difficult part …

AdditivePhylogeny

AdditivePhylogeny(*D*):

1. Pick an arbitrary leaf *j*.
2. Compute its limb length, *LimbLength*(*j*).
3. Subtract *LimbLength*(*j*) from each row and column to produce D^{bald} in which *j* is a bald limb (length 0).
4. Remove the *j*-th row and column of the matrix to form the $(n - 1) \times (n - 1)$ matrix D^{trim}.
5. Recursively call **AdditivePhylogeny(**D^{trim}**)** to obtain *Tree*(D^{trim}).
6. **Identify the point in *Tree*(D^{trim}) where leaf *j* should be attached.**
7. Attach *j* by an edge of length *LimbLength*(*j*) in order to form *Tree*(*D*).

Attaching a leaf to the tree after the recursive step is the difficult part …

… and what do we do about *non-additive* matrices?

Least-Squares Phylogeny

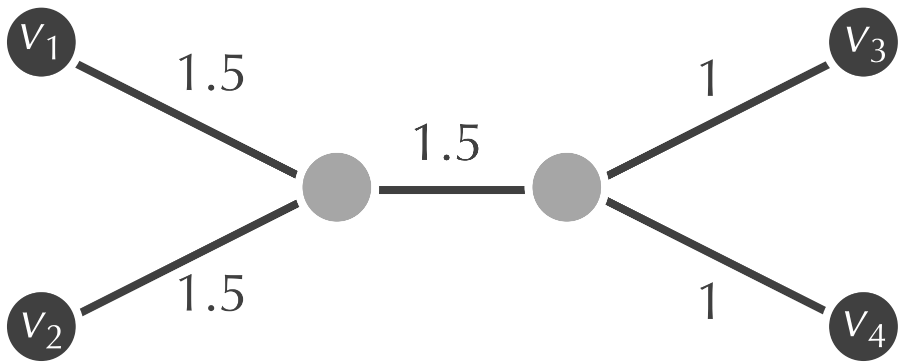
Idea: Maybe we can find a tree that fits a non-additive matrix *approximately*.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Least-Squares Phylogeny

Idea: Maybe we can find a tree that fits a non-additive matrix *approximately*.

T

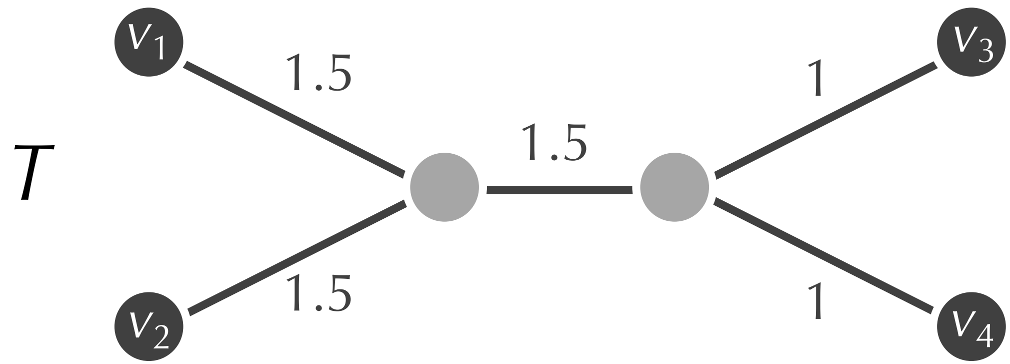


D

	v_1	v_2	v_3	v_4
v_1	0	3	4	3
v_2	3	0	4	5
v_3	4	4	0	2
v_4	3	5	2	0

Least-Squares Phylogeny

Idea: Maybe we can find a tree that fits a non-additive matrix *approximately*.



D

	v_1	v_2	v_3	v_4
v_1	0	3	4	3
v_2	3	0	4	5
v_3	4	4	0	2
v_4	3	5	2	0

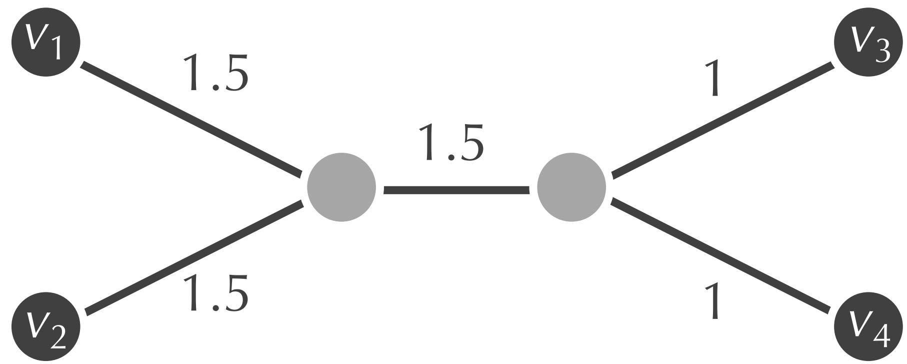
d

	v_1	v_2	v_3	v_4
v_1	0	3	4	4
v_2	3	0	4	4
v_3	4	4	0	2
v_4	4	4	2	0

Least-Squares Phylogeny

Idea: Maybe we can find a tree that fits a non-additive matrix *approximately*.

T



D

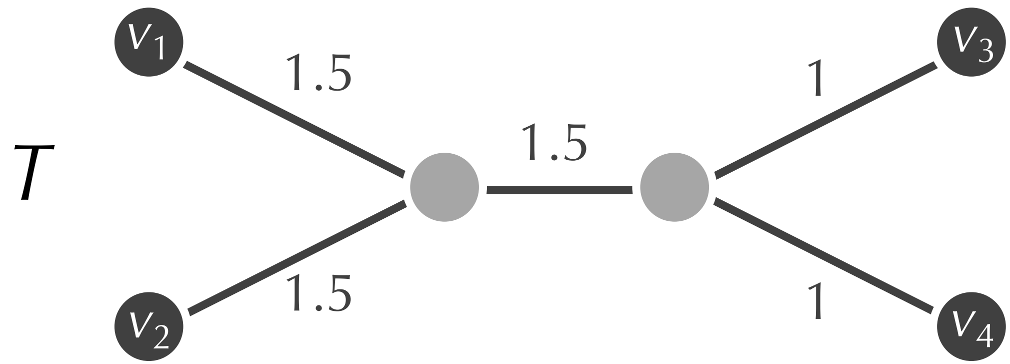
	v_1	v_2	v_3	v_4
v_1	0	3	4	**3**
v_2	3	0	4	**5**
v_3	4	4	0	2
v_4	3	5	2	0

d

	v_1	v_2	v_3	v_4
v_1	0	3	4	**4**
v_2	3	0	4	**4**
v_3	4	4	0	2
v_4	4	4	2	0

Least-Squares Phylogeny

$$Discrepancy(T, D) = \Sigma_{1 \leq i < j \leq n} (d_{i,j}(T) - D_{i,j})^2$$



D

	v_1	v_2	v_3	v_4
v_1	0	3	4	**3**
v_2	3	0	4	**5**
v_3	4	4	0	2
v_4	3	5	2	0

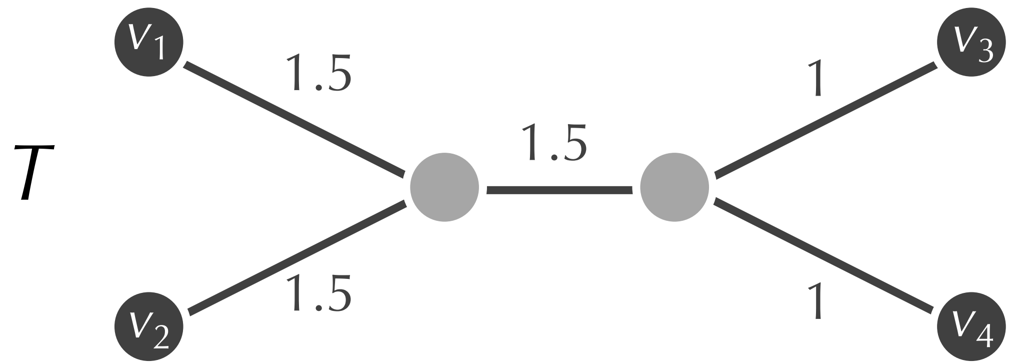
d

	v_1	v_2	v_3	v_4
v_1	0	3	4	**4**
v_2	3	0	4	**4**
v_3	4	4	0	2
v_4	4	4	2	0

Least-Squares Phylogeny

$$Discrepancy(T, D) = \Sigma_{1 \leq i < j \leq n} (d_{i,j}(T) - D_{i,j})^2$$

$$= 1^2 + 1^2 = 2$$



D

	v_1	v_2	v_3	v_4
v_1	0	3	4	**3**
v_2	3	0	4	**5**
v_3	4	4	0	2
v_4	3	5	2	0

d

	v_1	v_2	v_3	v_4
v_1	0	3	4	**4**
v_2	3	0	4	**4**
v_3	4	4	0	2
v_4	4	4	2	0

Least-Squares Phylogeny

Least-Squares Distance-Based Phylogeny Problem:
Given a distance matrix, find the tree that minimizes the sum of squared errors.

- **Input:** An n x n distance matrix D.
- **Output:** A weighted tree T with n leaves minimizing *Discrepancy*(T, D) over all weighted trees with n leaves.

Least-Squares Phylogeny

Least-Squares Distance-Based Phylogeny Problem:
Given a distance matrix, find the tree that minimizes the sum of squared errors.

- **Input:** An n x n distance matrix D.
- **Output:** A weighted tree T with n leaves minimizing *Discrepancy*(T, D) over all weighted trees with n leaves.

Unfortunately, this problem is *NP*-Complete...

The Critical Insight

Our first idea of using neighbors to construct the tree was a good one! The problem was not with our idea, but with the distance matrix itself!

The Neighbor-Joining Theorem

Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n - 2) \bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

where $TotalDistance_D(i)$ is the sum of distances from i to all other leaves.

The Neighbor-Joining Theorem

Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n-2)\bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

where $TotalDistance_D(i)$ is the sum of distances from i to all other leaves.

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

The Neighbor-Joining Theorem

Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n - 2) \bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

where $TotalDistance_D(i)$ is the sum of distances from i to all other leaves.

D

	v_1	v_2	v_3	v_4	$TotalDistance_D$
v_1	0	13	21	22	56
v_2	13	0	12	13	38
v_3	21	12	0	13	46
v_4	22	13	13	0	48

The Neighbor-Joining Theorem

Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n-2) \bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

where $TotalDistance_D(i)$ is the sum of distances from i to all other leaves.

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

$TotalDistance_D$
56
38
46
48

D^*

	v_1	v_2	v_3	v_4
v_1	0	-68	-60	-60
v_2	-68	0	-60	-60
v_3	-60	-60	0	-68
v_4	-60	-60	-68	0

The Neighbor-Joining Theorem

Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n - 2) \bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

where $TotalDistance_D(i)$ is the sum of distances from i to all other leaves.

D

	v_1	v_2	v_3	v_4	$TotalDistance_D$
v_1	0	13	21	22	56
v_2	13	0	12	13	38
v_3	21	12	0	13	46
v_4	22	13	13	0	48

D^*

	v_1	v_2	v_3	v_4
v_1	0	-68	-60	-60
v_2	-68	0	-60	-60
v_3	-60	-60	0	-68
v_4	-60	-60	-68	0

The Neighbor-Joining Theorem

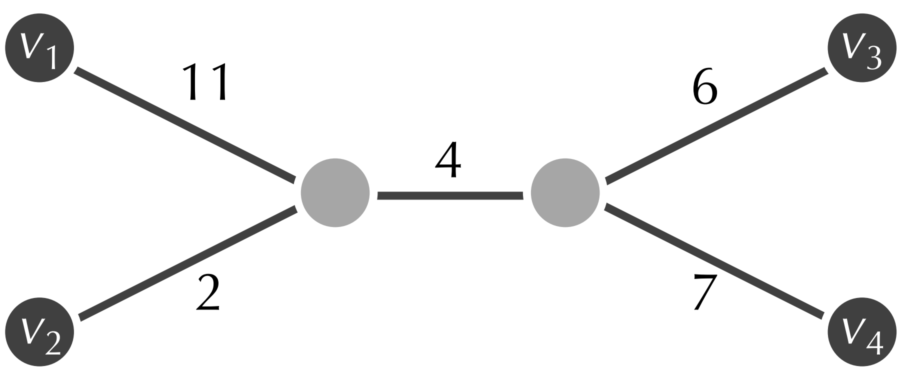
Given an n x n distance matrix D, its **neighbor-joining matrix** is the matrix D^* defined as

$$D^*_{i,j} = (n - 2) \bullet D_{i,j} - TotalDistance_D(i) - TotalDistance_D(j)$$

Note: What does D^* do to outliers?

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0



The Neighbor-Joining Theorem

Neighbor-Joining Theorem: If D is additive, then the smallest element of D^* corresponds to neighboring leaves in $Tree(D)$!

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	12	13
v_3	21	12	0	13
v_4	22	13	13	0

$TotalDistance_D$
56
38
46
48

D^*

	v_1	v_2	v_3	v_4
v_1	0	-68	-60	-60
v_2	-68	0	-60	-60
v_3	-60	-60	0	-68
v_4	-60	-60	-68	0

The Neighbor-Joining Theorem

Neighbor-Joining Theorem: If D is additive, then the smallest element of D^* corresponds to neighboring leaves in *Tree*(D)!

D

	v_1	v_2	v_3	v_4
v_1	0	13	21	22
v_2	13	0	**12**	13
v_3	21	**12**	0	13
v_4	22	13	13	0

$TotalDistance_D$
56
38
46
48

D^*

	v_1	v_2	v_3	v_4
v_1	0	**-68**	-60	-60
v_2	**-68**	0	-60	-60
v_3	-60	-60	0	**-68**
v_4	-60	-60	**-68**	0

Neighbor-Joining in Action

D^*	v_1	v_2	v_3	v_4	$TotalDistance_D$
v_1	0	-68	-60	-60	56
v_2	-68	0	-60	-60	38
v_3	-60	-60	0	-68	46
v_4	-60	-60	-68	0	48

1. Construct neighbor-joining matrix D^* from D.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

D^*	v_1	v_2	v_3	v_4	$TotalDistance_D$
v_1	0	**-68**	-60	-60	56
v_2	**-68**	0	-60	-60	38
v_3	-60	-60	0	**-68**	46
v_4	-60	-60	**-68**	0	48

2. Find a minimum element $D^*_{i,j}$ of D^*.

Neighbor-Joining in Action

$$D^* = \begin{array}{c|cccc|c} & v_1 & v_2 & v_3 & v_4 & TotalDistance_D \\ \hline v_1 & 0 & \mathbf{-68} & -60 & -60 & 56 \\ v_2 & \mathbf{-68} & 0 & -60 & -60 & 38 \\ v_3 & -60 & -60 & 0 & -68 & 46 \\ v_4 & -60 & -60 & -68 & 0 & 48 \end{array}$$

2. Find a minimum element $D^*_{i,j}$ of D^*.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

		v_1	v_2	v_3	v_4	$TotalDistance_D$
	v_1	0	**-68**	-60	-60	**56**
D^*	v_2	**-68**	0	-60	-60	**38**
	v_3	-60	-60	0	-68	46
	v_4	-60	-60	-68	0	48

$$\Delta_{i,j} = (\mathbf{56} - \mathbf{38}) / (4 - 2)$$
$$= 9$$

3. Compute $\Delta_{i,j} = (TotalDistance_D(i) - TotalDistance_D(j)) / (n - 2)$.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

D

	v_1	v_2	v_3	v_4	$TotalDistance_D$
v_1	0	**13**	21	22	56
v_2	**13**	0	12	13	38
v_3	21	12	0	13	46
v_4	22	13	13	0	48

$$\Delta_{i,j} = (56 - 38) / (4 - 2)$$
$$= 9$$

$$LimbLength(i) = ½(13 + 9) = 11$$
$$LimbLength(i) = ½(13 - 9) = 2$$

4. Set $LimbLength(i)$ equal to $½(D_{i,j} + \Delta_{i,j})$ and $LimbLength(j)$ equal to $½(D_{i,j} - \Delta_{i,j})$.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

D

	v_1	v_2	v_3	v_4	*TotalDistance*$_D$
v_1	0	**13**	21	22	56
v_2	**13**	0	12	13	38
v_3	21	12	0	13	46
v_4	22	13	13	0	48

$\Delta_{i,j} = (56 - 38) / (4 - 2)$
$= 9$

LimbLength(i) = ½(13 + 9) = 11
LimbLength(i) = ½(13 – 9) = 2

4. Set *LimbLength*(i) equal to $½(D_{i,j} + \Delta_{i,j})$ and *LimbLength*(j) equal to $½(D_{i,j} - \Delta_{i,j})$.

Wait *… where do these formulas come from?*

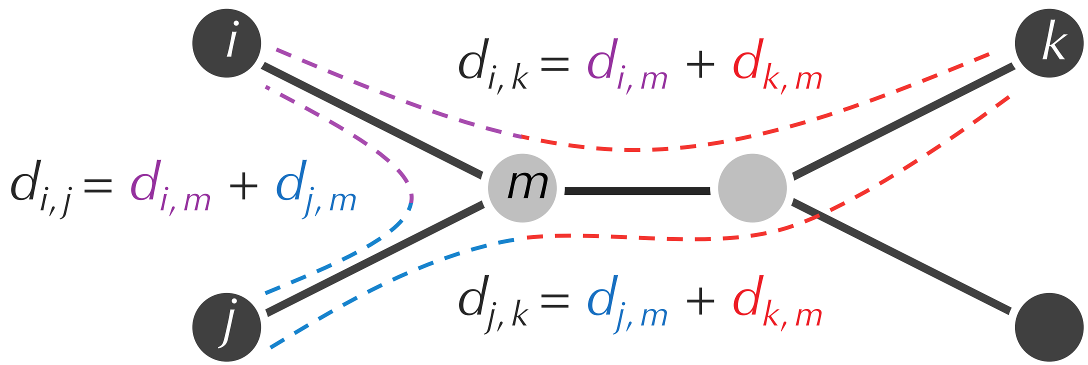
Neighbor-Joining in Action

D'

	m	$\boldsymbol{v_3}$	$\boldsymbol{v_4}$	***TotalDistance***$_D$
m	0	10	11	21
$\boldsymbol{v_3}$	10	0	13	23
$\boldsymbol{v_4}$	11	13	0	24

5. Form a matrix D' by removing i-th and j-th row/column from D and adding an m-th row/column such that for any k, $D_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$.

Flashback: Computation of $d_{k,m}$



$$d_{k,m} = [(d_{i,m} + d_{k,m}) + (d_{j,m} + d_{k,m}) - (d_{i,m} + d_{j,m})] / 2$$

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

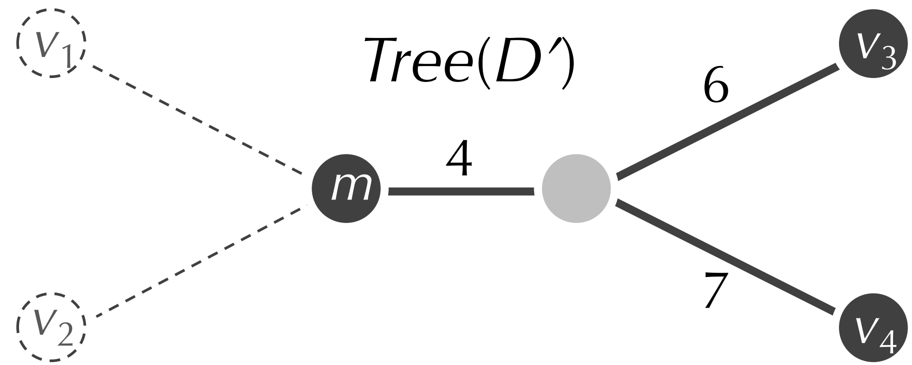
$$d_{k,m} = (D_{i,k} + D_{j,k} - D_{i,j}) / 2$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

D'

	m	v_3	v_4
m	0	10	11
v_3	10	0	13
v_4	11	13	0



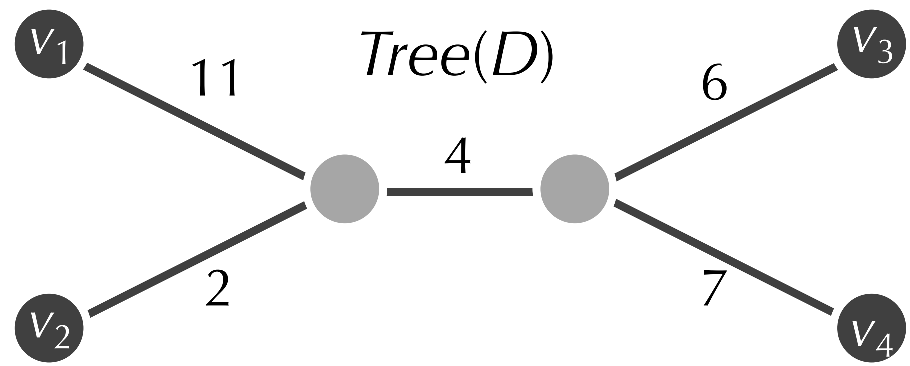
6. Apply **NeighborJoining** recursively to D' to obtain *Tree*(D').

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining in Action

D'

	$\boldsymbol{m}$	$\boldsymbol{v_3}$	$\boldsymbol{v_4}$
$\boldsymbol{m}$	0	10	11
$\boldsymbol{v_3}$	10	0	13
$\boldsymbol{v_4}$	11	13	0



$$LimbLength(v_1) = \tfrac{1}{2}(13 + 9) = 11$$
$$LimbLength(v_2) = \tfrac{1}{2}(13 - 9) = 2$$

7. Reattach limbs of *i* and *j* to obtain *Tree*(*D*).

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

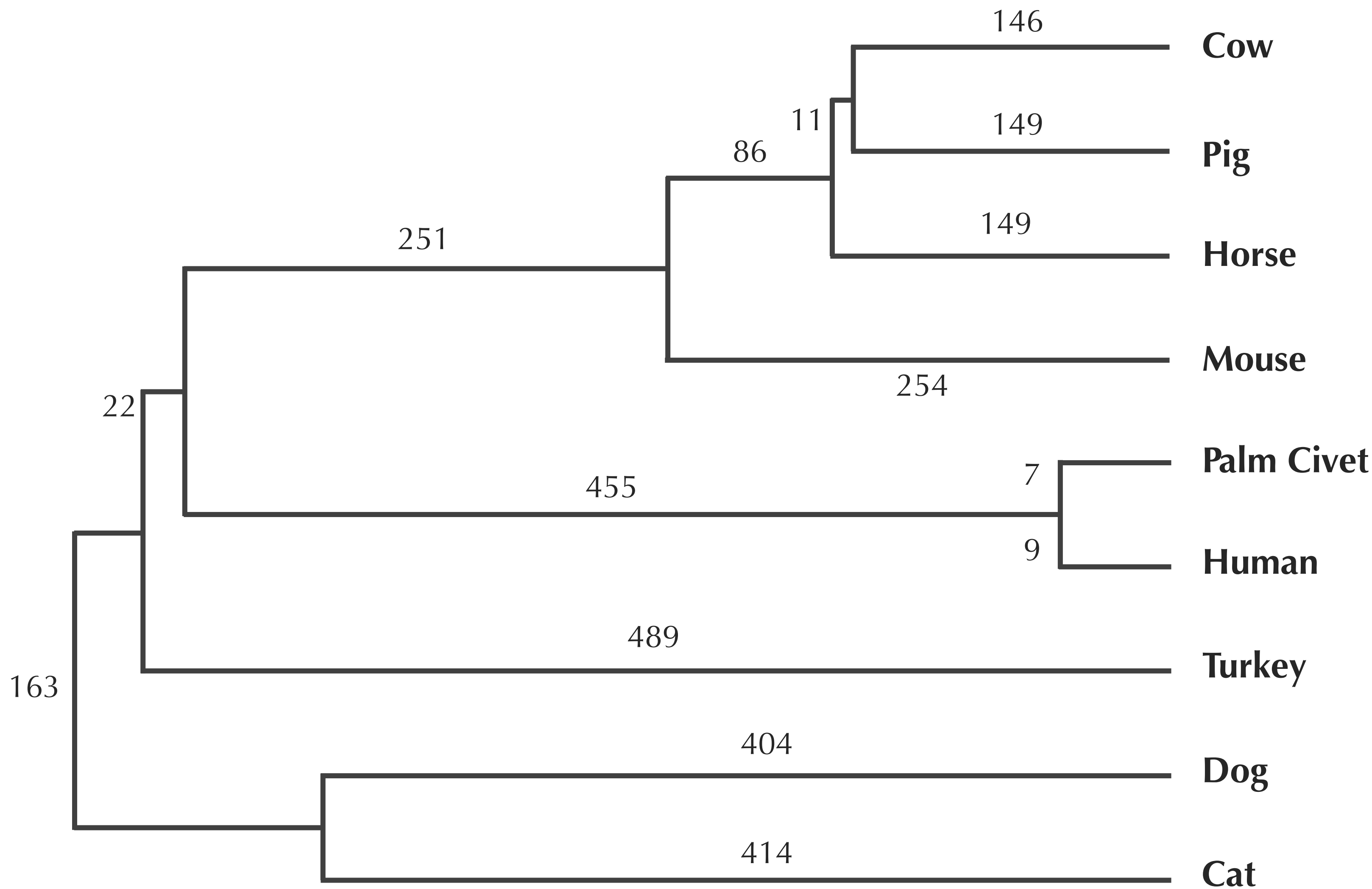
Neighbor-Joining in Action

NeighborJoining(D):

1. Construct neighbor-joining matrix D^* from D.
2. Find a minimum element $D^*_{i,j}$ of D^*.
3. Compute $\Delta_{i,j} = (TotalDistance_D(i) - TotalDistance_D(j)) / (n - 2)$.
4. Set $LimbLength(i)$ equal to $\frac{1}{2}(D_{i,j} + \Delta_{i,j})$ and $LimbLength(j)$ equal to $\frac{1}{2}(D_{i,j} - \Delta_{i,j})$.
5. Form a matrix D' by removing i-th and j-th row/column from D and adding an m-th row/column such that for any k, $D_{k,m} = (D_{k,i} + D_{k,j} - D_{i,j}) / 2$.
6. Apply **NeighborJoining** recursively to D' to obtain $Tree(D')$.
7. Reattach limbs of i and j to obtain $Tree(D)$.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

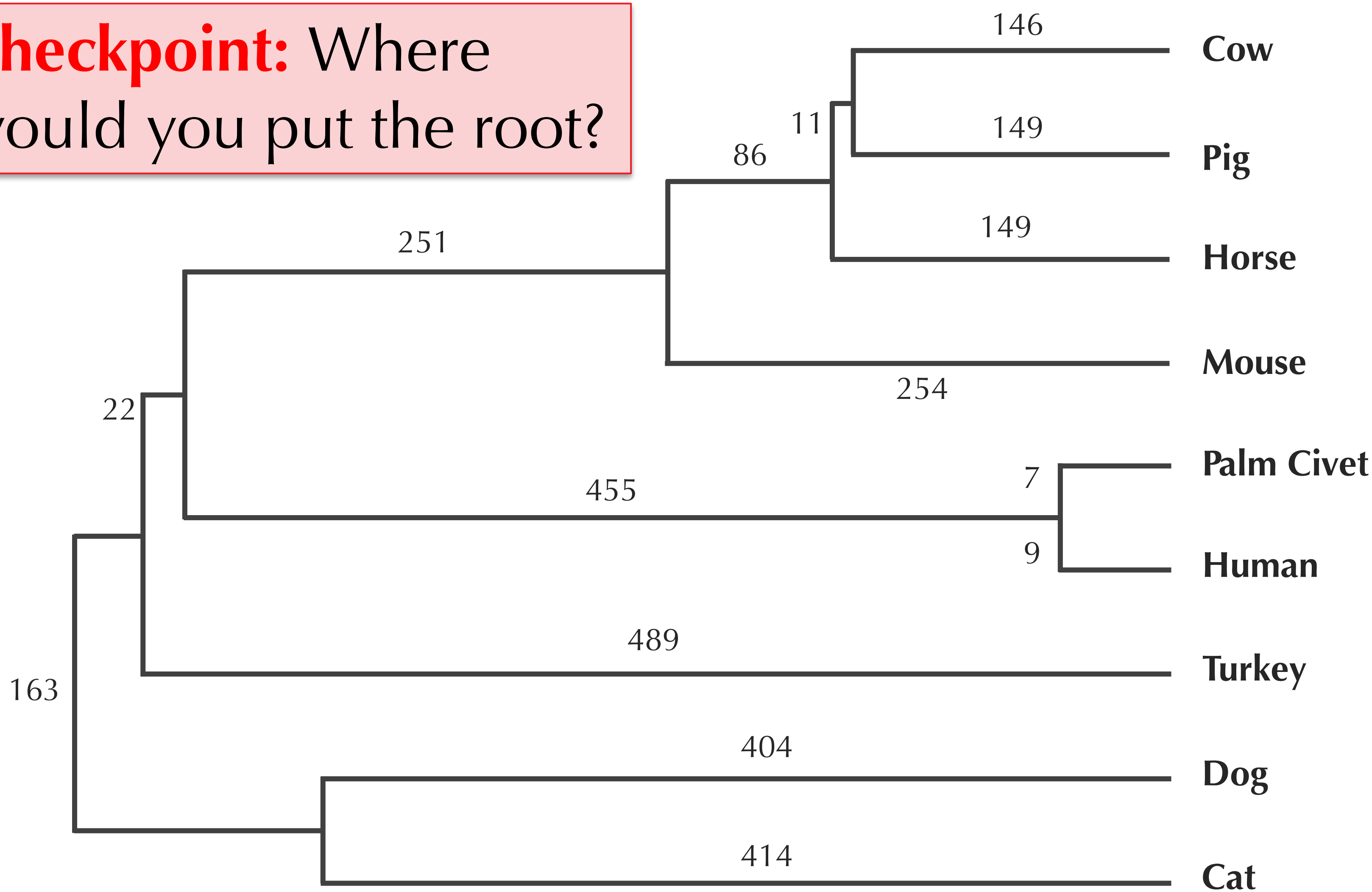
Neighbor-Joining on Coronavirus "Spike Proteins"



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining on Coronavirus "Spike Proteins"

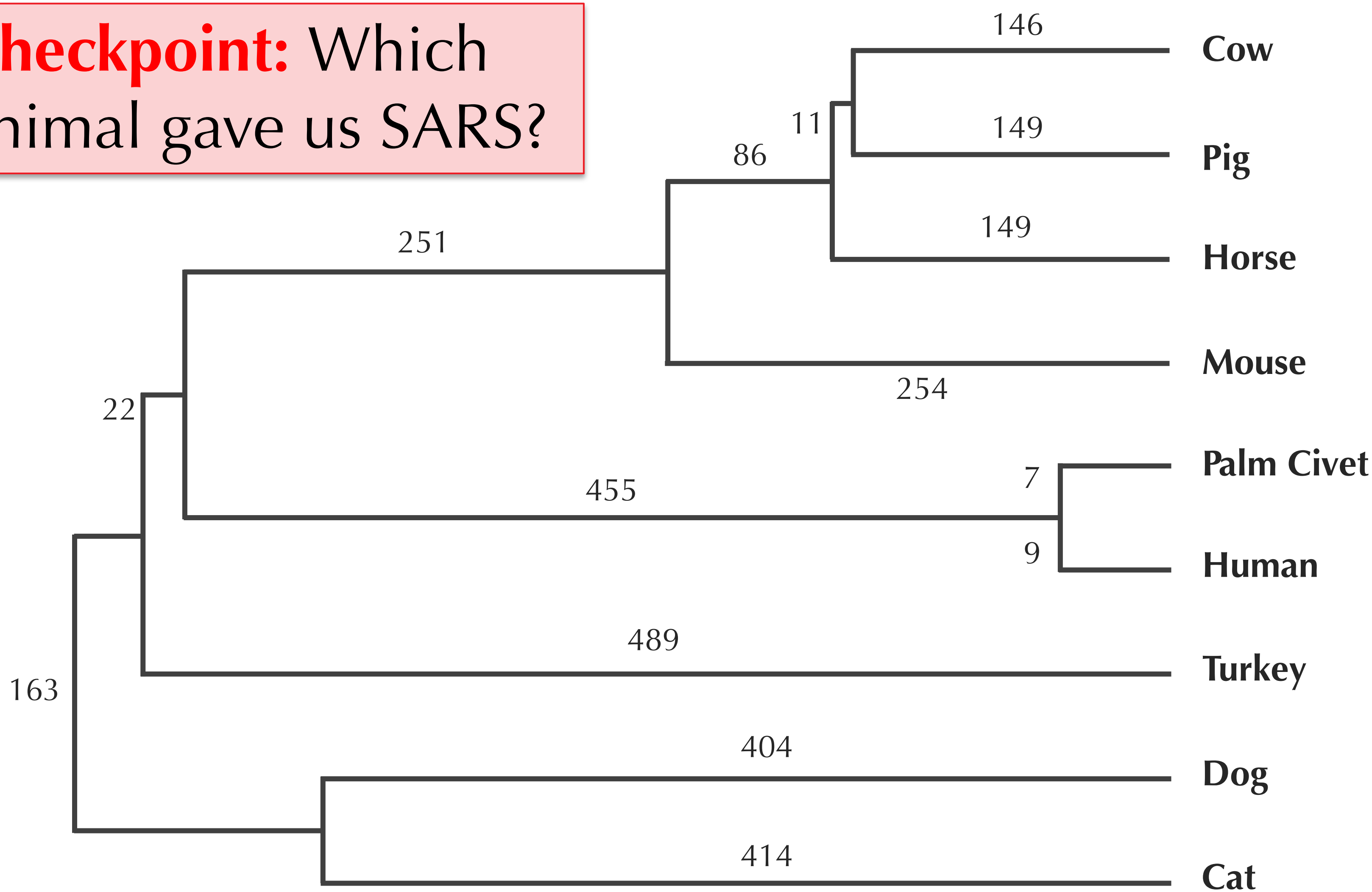
Checkpoint: Where would you put the root?



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

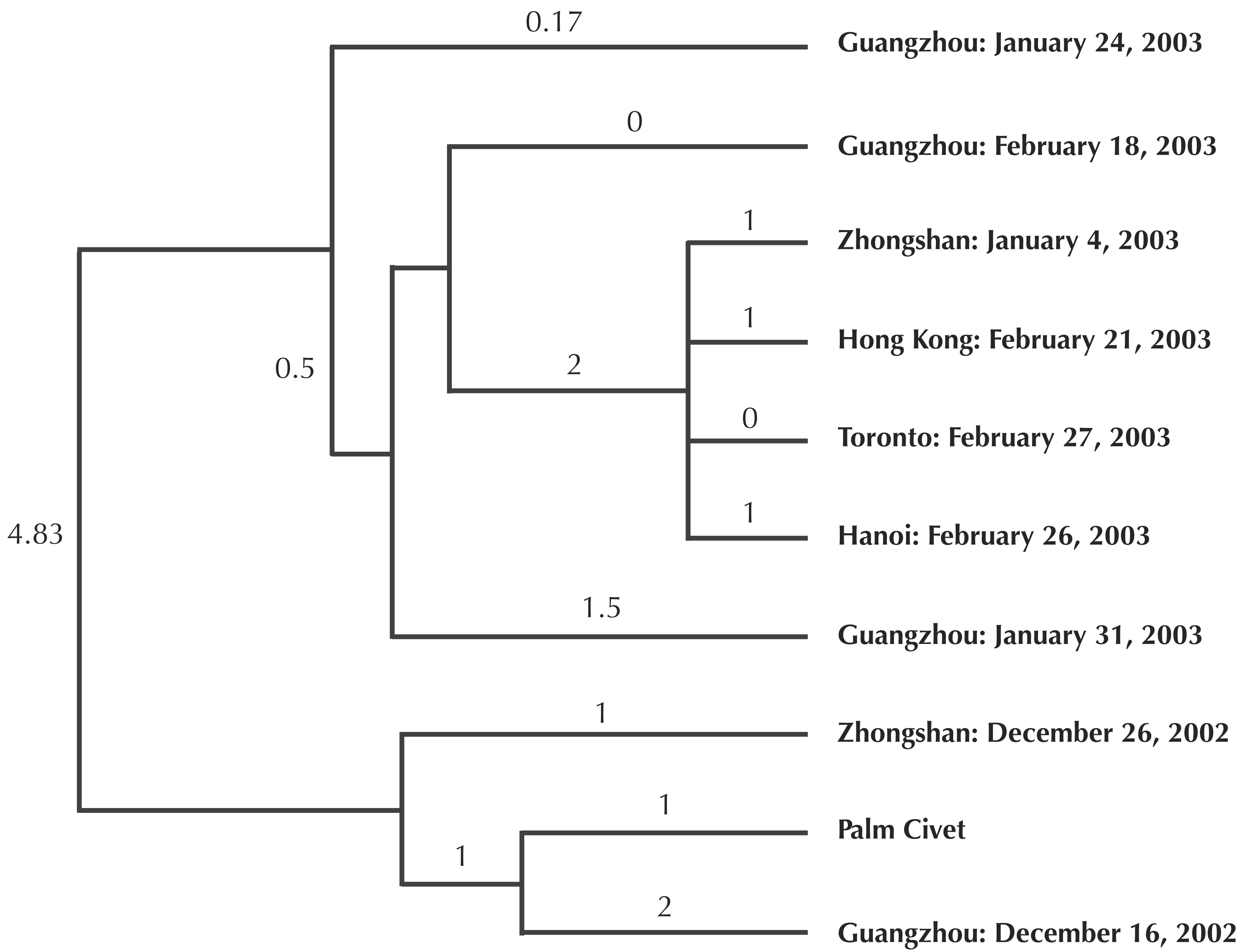
Neighbor-Joining on Coronavirus "Spike Proteins"

Checkpoint: Which animal gave us SARS?



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Neighbor-Joining on Coronaviruses



Weakness of Distance-Based Methods

Distance-based algorithms for evolutionary tree reconstruction say nothing about ancestral states at internal nodes.

Weakness of Distance-Based Methods

Distance-based algorithms for evolutionary tree reconstruction say nothing about ancestral states at internal nodes.

We *lost* information when we converted a multiple alignment to a distance matrix...

SPECIES	ALIGNMENT	DISTANCE MATRIX			
		Chimp	**Human**	**Seal**	**Whale**
Chimp	`ACGTAGGCCT`	0	3	6	4
Human	`ATGTAAGACT`	3	0	7	5
Seal	`TCGAGAGCAC`	6	7	0	2
Whale	`TCGAAAGCAT`	4	5	2	0

Which Animal Gave Us SARS?

Evolutionary Trees Part 2: The Small Parsimony Algorithm for Inferring Ancestral States

Phillip Compeau and Pavel Pevzner

Bioinformatics Algorithms: An Active Learning Approach

©2018 by Compeau and Pevzner. All rights reserved.


What is the main evolutionary division of dinosaurs?
Courtesy: Universal Studios

Character-Based Construction

Biologists used to form trees based on anatomical or physiological properties called **characters**.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Character-Based Construction

Biologists used to form trees based on anatomical or physiological properties called **characters**.

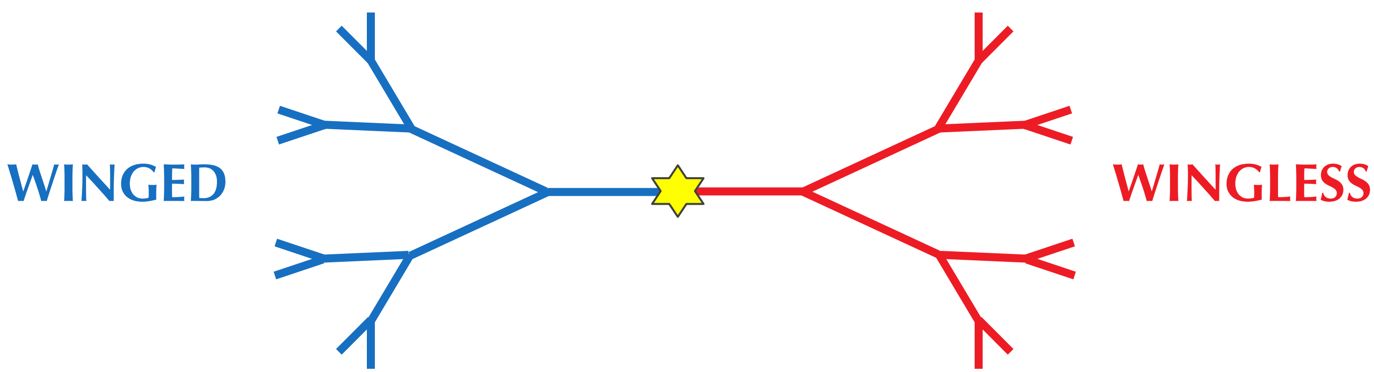
Checkpoint: Say you wanted to construct an evolutionary tree of all insects. What might be the first thing you would do?

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Character-Based Construction

Biologists used to form trees based on anatomical or physiological properties called **characters**.

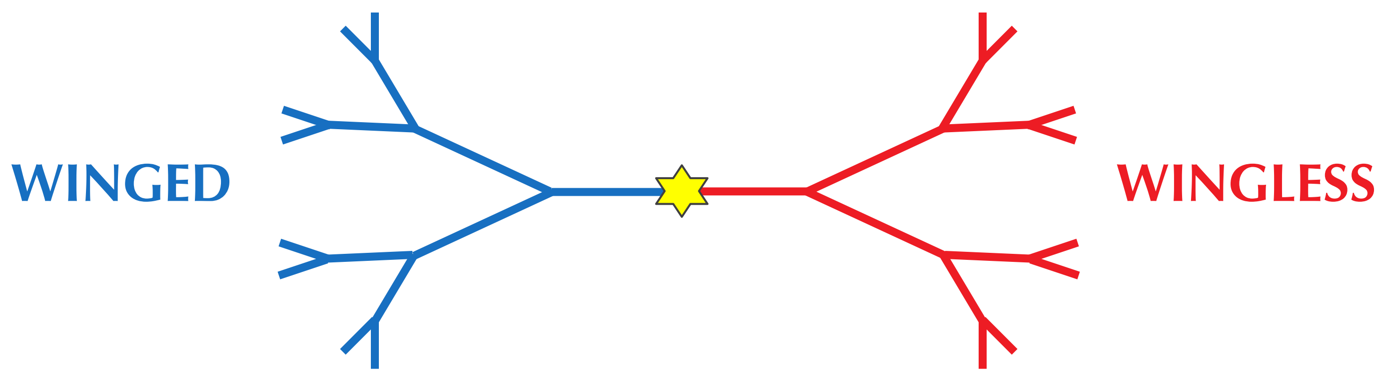
Checkpoint: Say you wanted to construct an evolutionary tree of all insects. What might be the first thing you would do?



Character-Based Construction

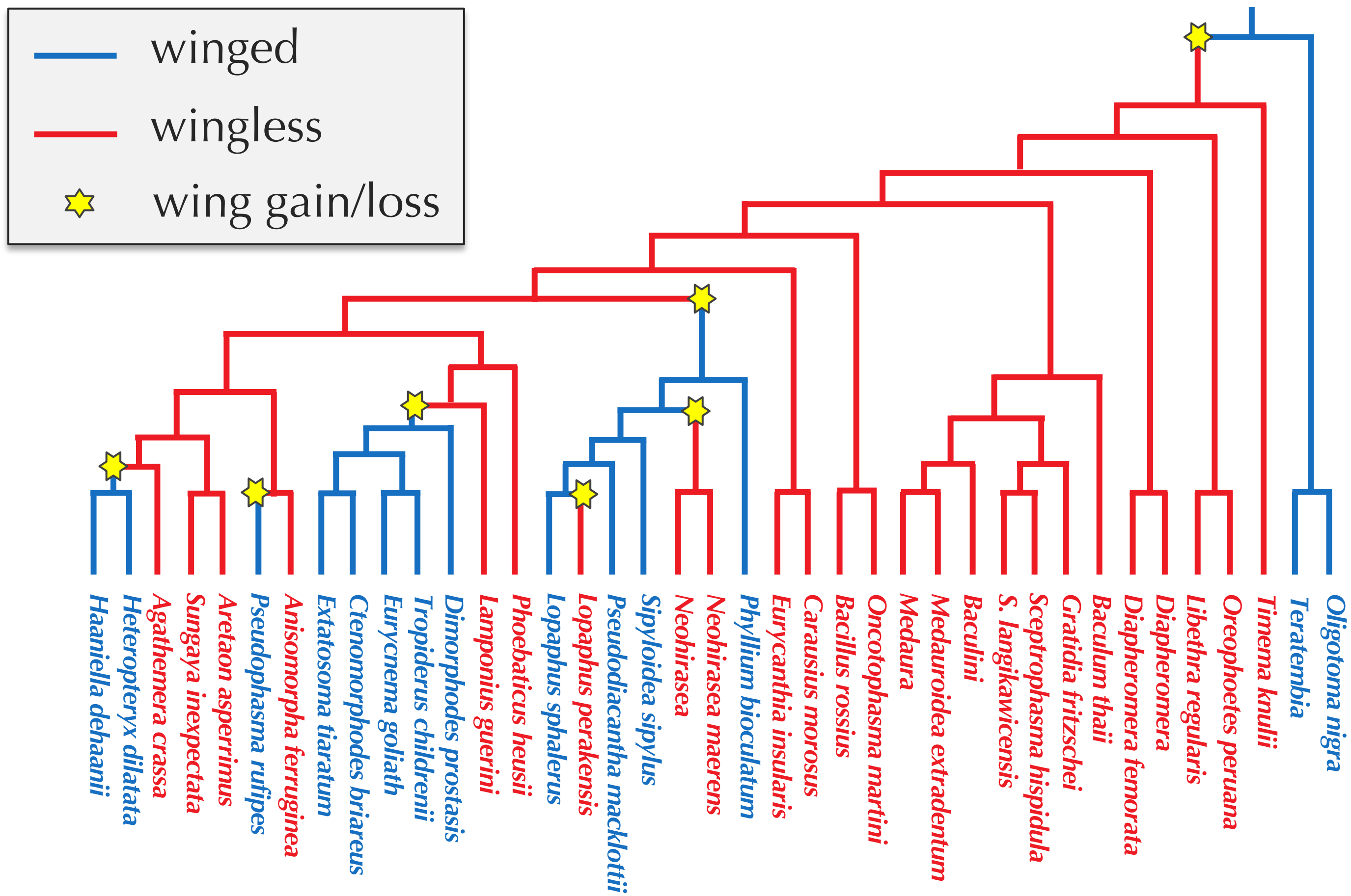
Dollo's principle of irreversibility (1893): evolution doesn't reinvent the same organ (e.g. insect wings).

Checkpoint: Say you wanted to construct an evolutionary tree of all insects. What might be the first thing you would do?



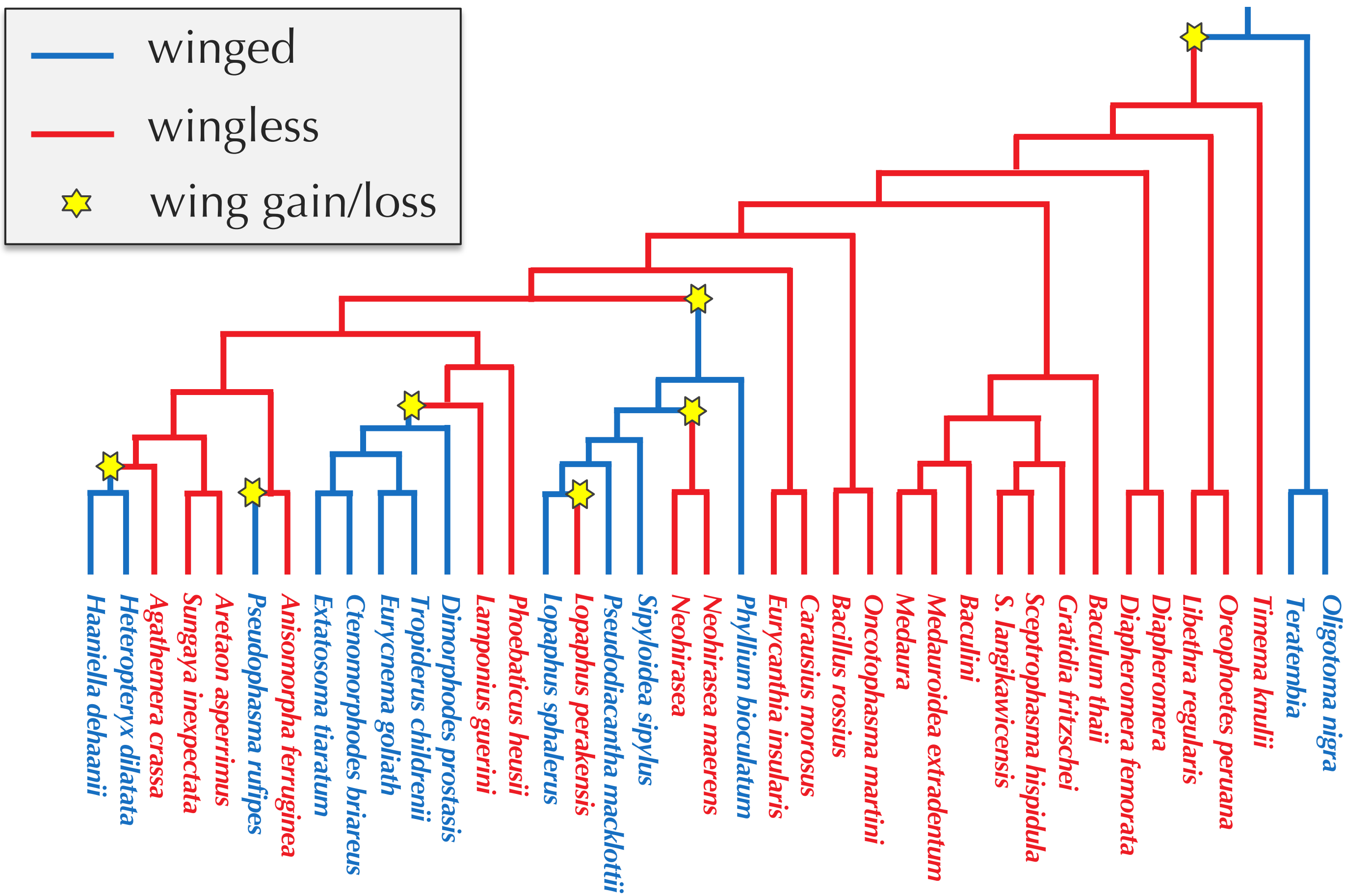
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Dollo's Principle Violated in Stick Insect Phylogeny



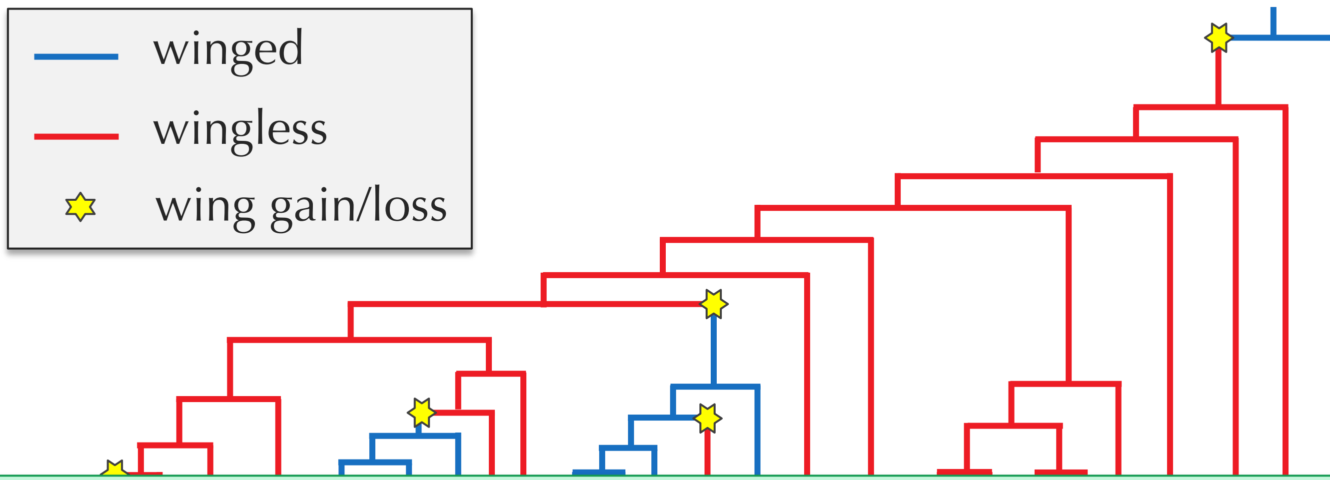
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Dollo's Principle Violated in Stick Insect Phylogeny



Checkpoint: What do you think happened?

Dollo's Principle Violated in Stick Insect Phylogeny



Key Point: This approach won't help reconstruct the tree *structure*, but perhaps we can infer *ancestral* characters after constructing a tree using another method (e.g., neighbor-joining).

Checkpoint: What do you think happened?

An Alignment Is a Character Table

Chimp	ACGTAGGCCT
Human	ATGTAAGACT
Seal	TCGAGAGCAC
Whale	TCGAAAGCAT

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

An Alignment Is a Character Table

Chimp	ACGTAGGCCT
Human	ATGTAAGACT
Seal	TCGAGAGCAC
Whale	TCGAAAGCAT

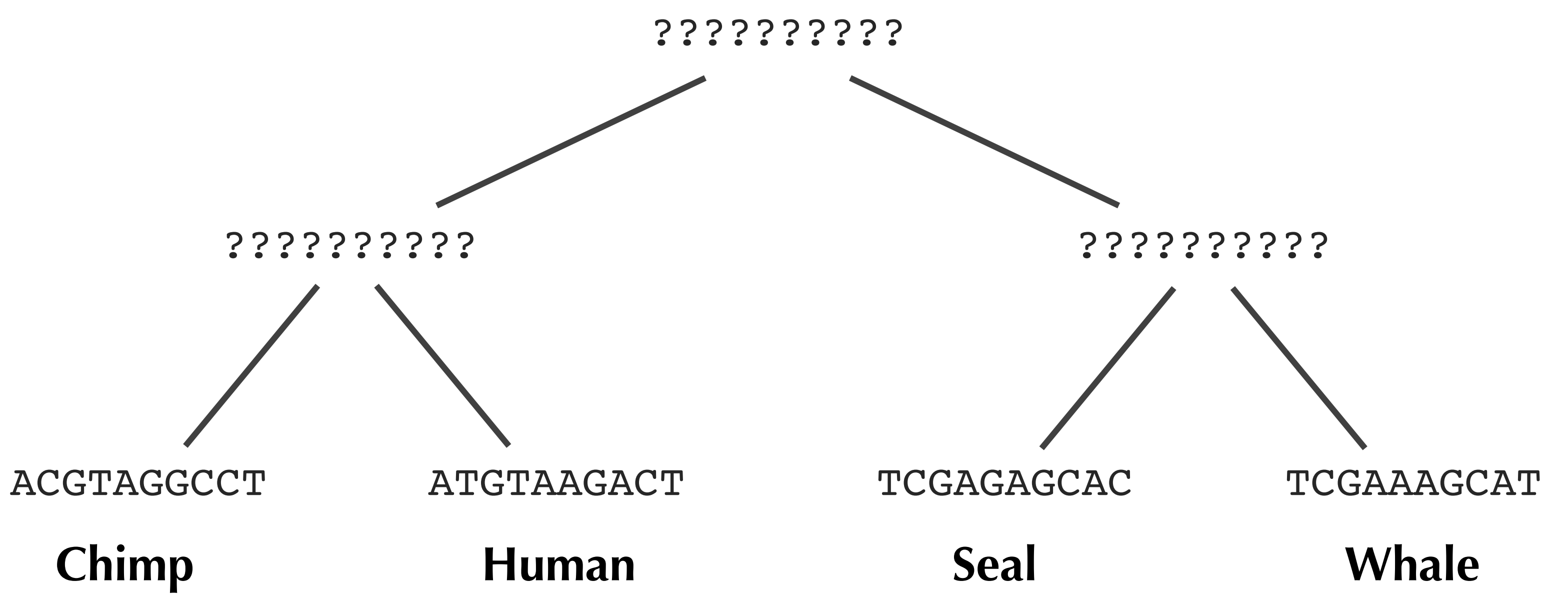
n species

m characters

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

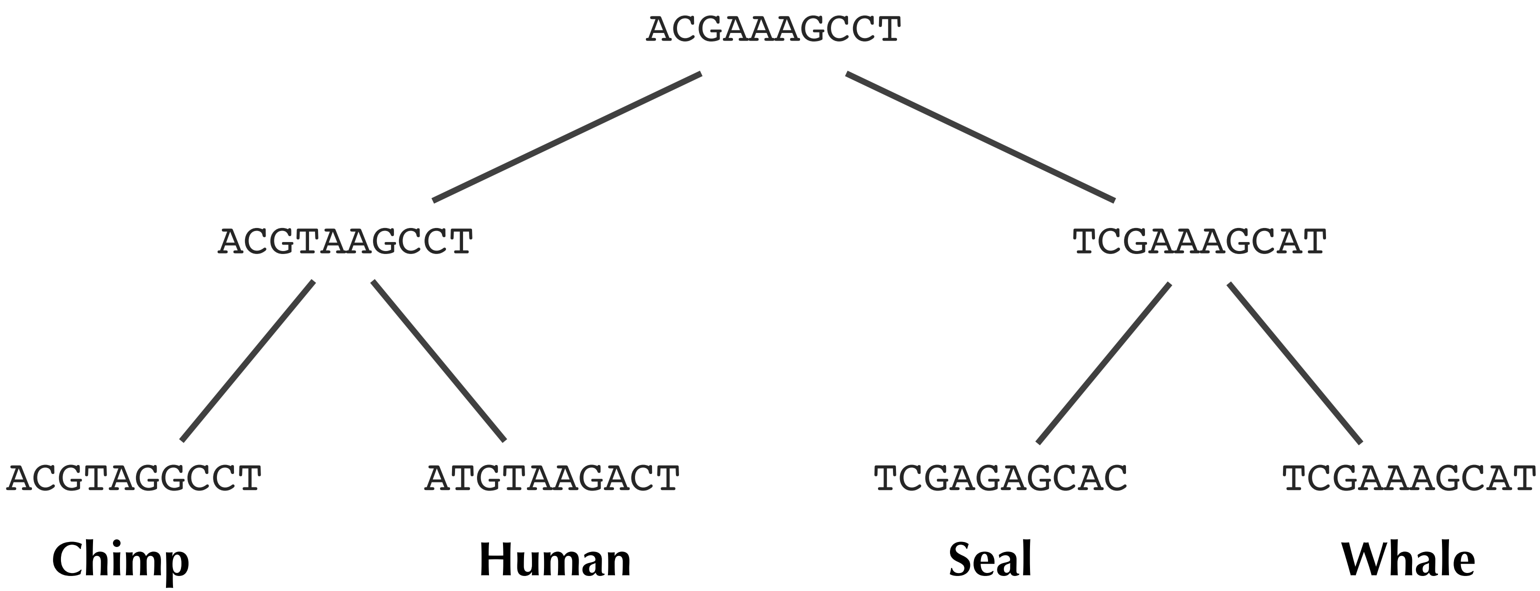
Toward a Computational Problem

Chimp	ACGTAGGCCT
Human	ATGTAAGACT
Seal	TCGAGAGCAC
Whale	TCGAAAGCAT



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

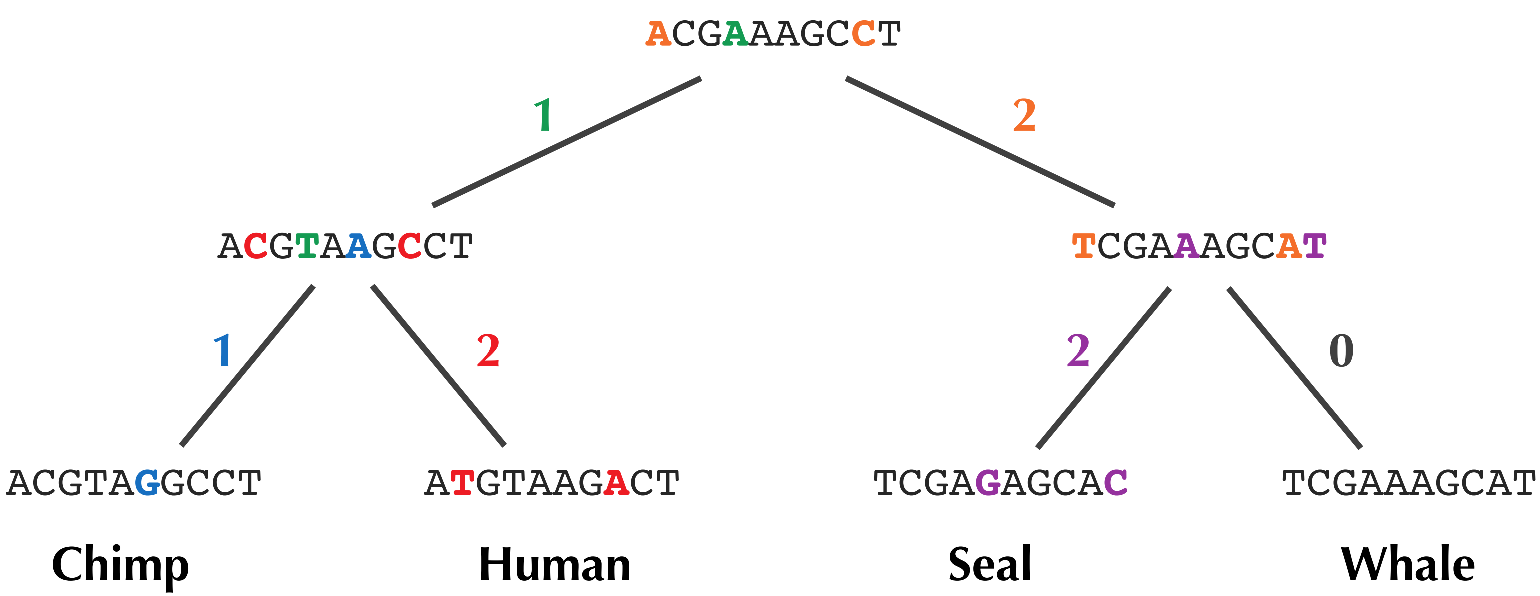
Toward a Computational Problem



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

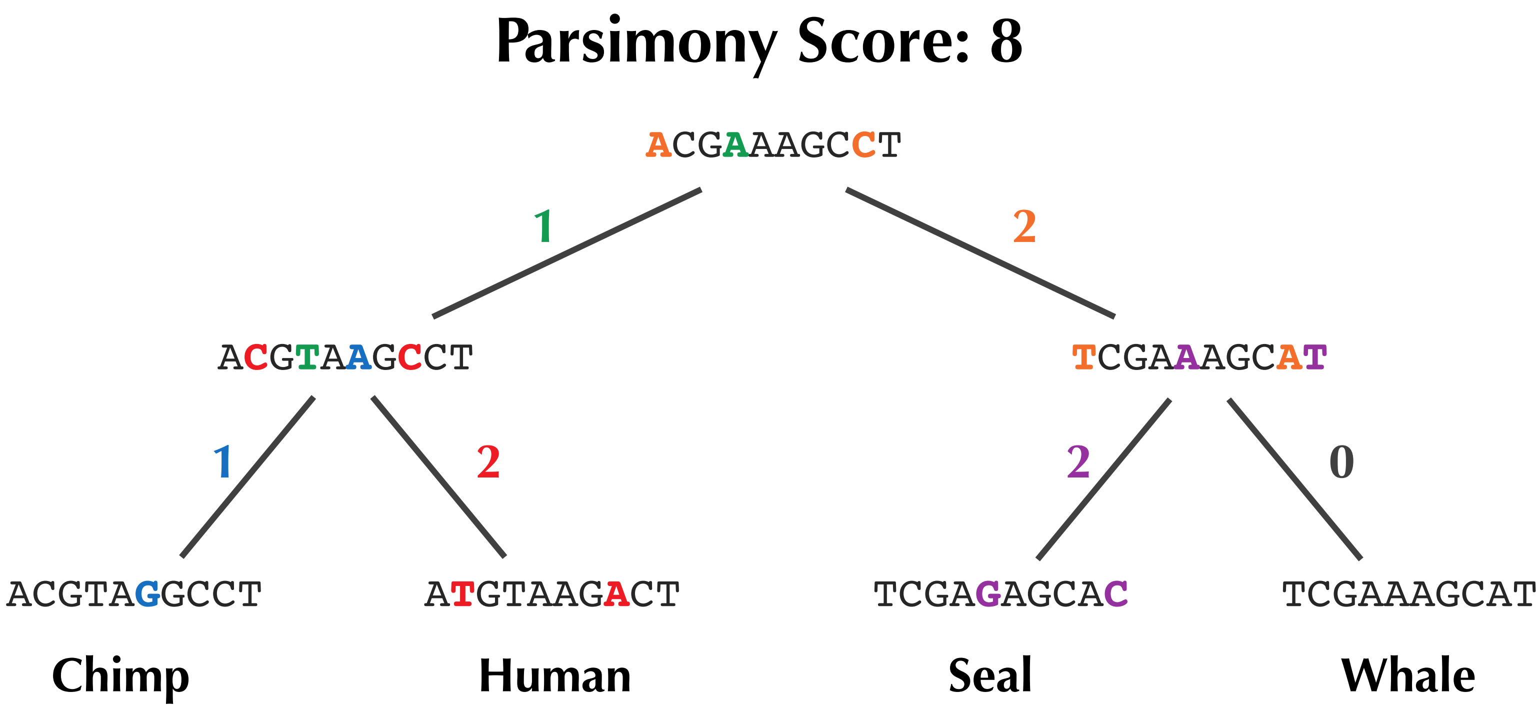
Parsimony score: sum of Hamming distances (total mismatches) along each edge.



Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Parsimony score: sum of Hamming distances (total mismatches) along each edge.

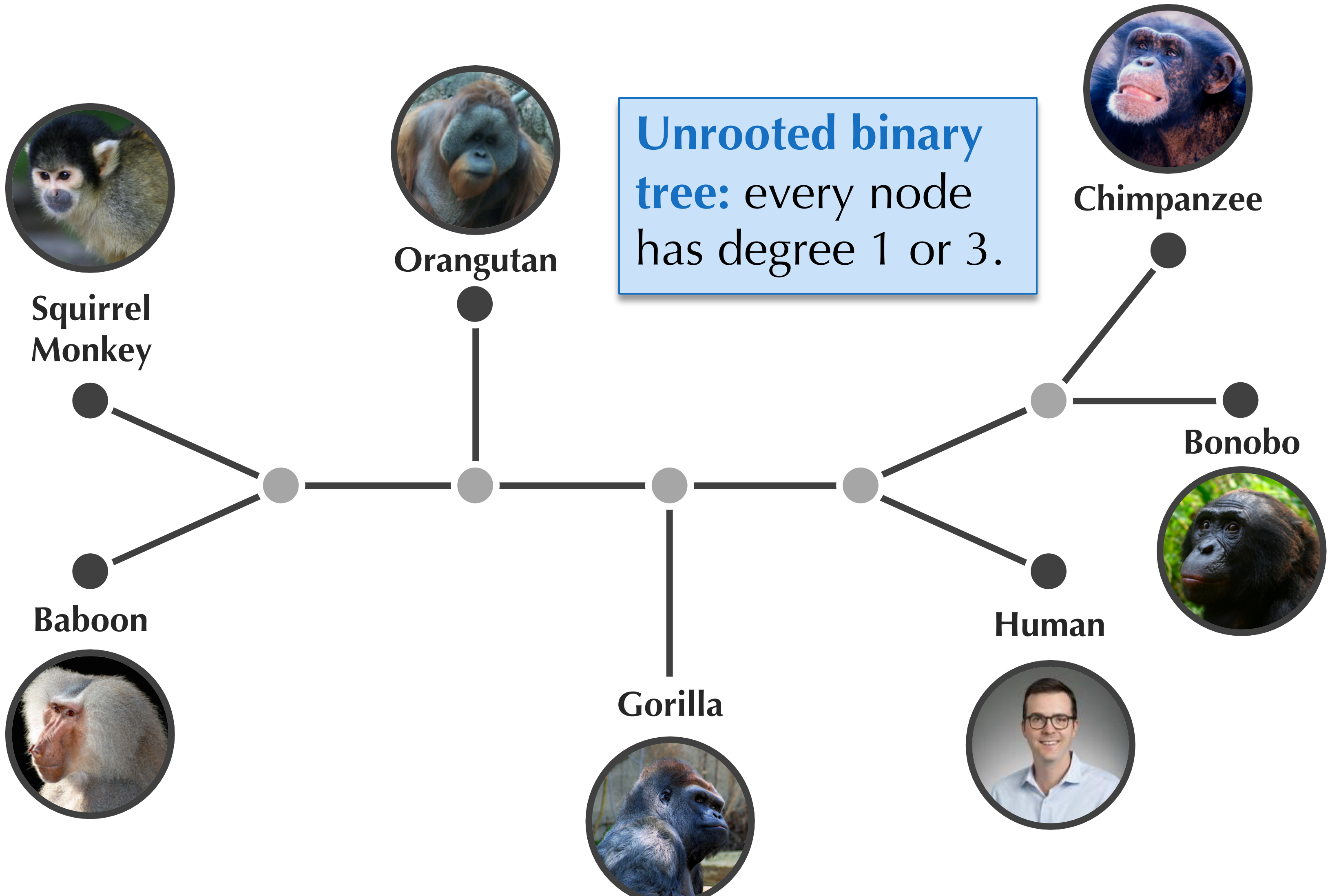


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Modeling Speciations

Researchers often assume that all internal nodes correspond to **speciations**, where one species splits into two.

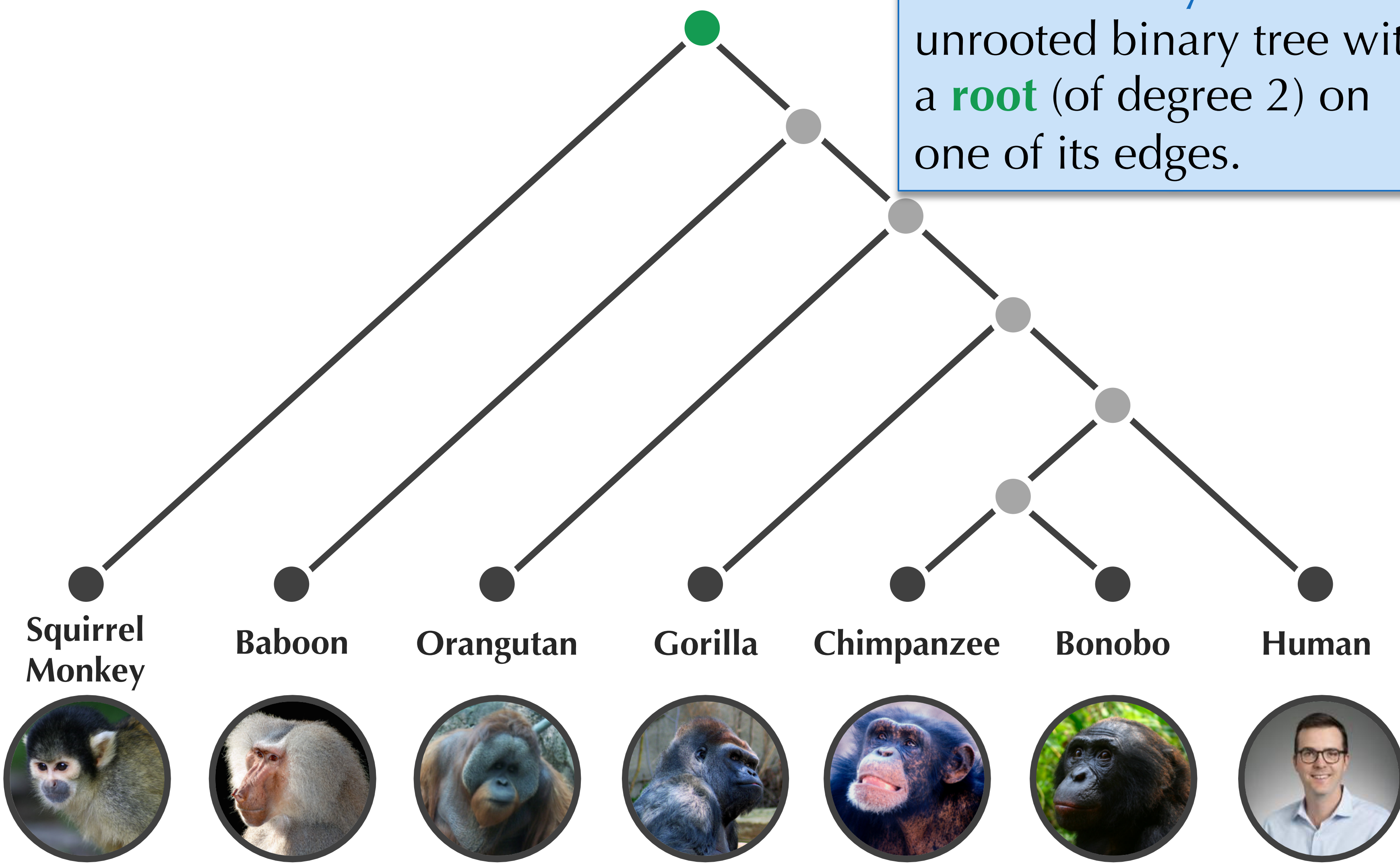
Modeling Speciations



Unrooted binary tree: every node has degree 1 or 3.

Modeling Speciations

Rooted binary tree: an unrooted binary tree with a **root** (of degree 2) on one of its edges.



Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

- **Input:** A rooted binary tree with each leaf labeled by a string of length *m*.
- **Output:** A labeling of all other nodes of the tree by strings of length *m* that minimizes the tree's parsimony score.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

- **Input:** A rooted binary tree with each leaf labeled by a string of length *m*.
- **Output:** A labeling of all other nodes of the tree by strings of length *m* that minimizes the tree's parsimony score.

Checkpoint: Is there any way we can simplify this problem statement?

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

- **Input:** A rooted binary tree with each leaf labeled by a **single symbol**.
- **Output:** A labeling of all other nodes of the tree by **single symbols** that minimizes the tree's parsimony score.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

- **Input:** A rooted binary tree with each leaf labeled by a **single symbol**.
- **Output:** A labeling of all other nodes of the tree by **single symbols** that minimizes the tree's parsimony score.

Checkpoint: Why is this an acceptable simplification?

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

- **Input:** A rooted binary tree with each leaf labeled by a **single symbol**.
- **Output:** A labeling of all other nodes of the tree by **single symbols** that minimizes the tree's parsimony score.

Answer: We may choose to assume that the characters are *independent*.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

Toward a Computational Problem

Small Parsimony Problem: *Find the most parsimonious labeling of the internal nodes of a rooted tree.*

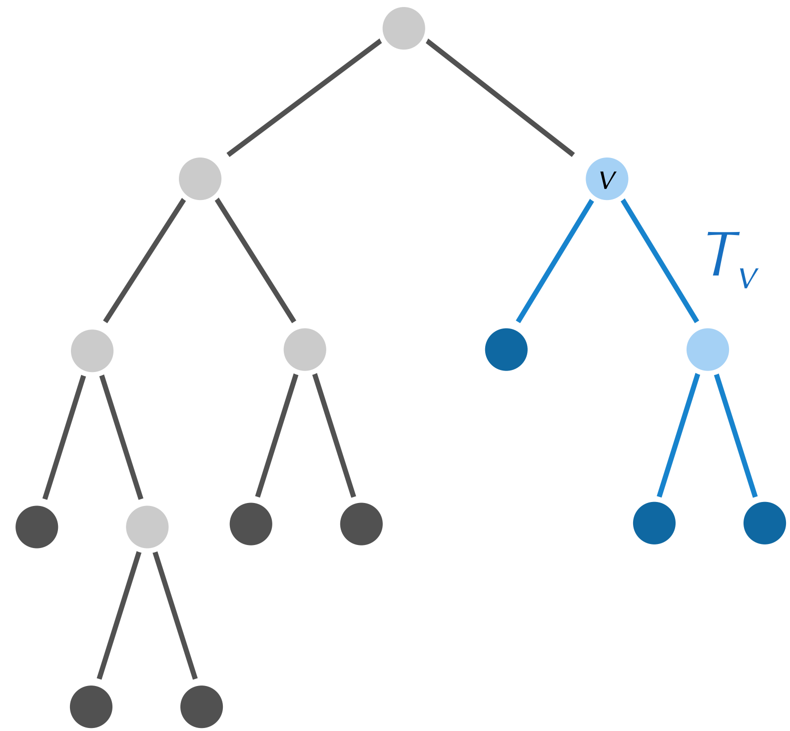
- **Input:** A rooted binary tree with each leaf labeled by a **single symbol**.
- **Output:** A labeling of all other nodes of the tree by **single symbols** that minimizes the tree's parsimony score.

Checkpoint: Any thoughts on what approach we might use to solve this problem?

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Dynamic Programming Algorithm

Let T_v denote the subtree of T whose root is v.

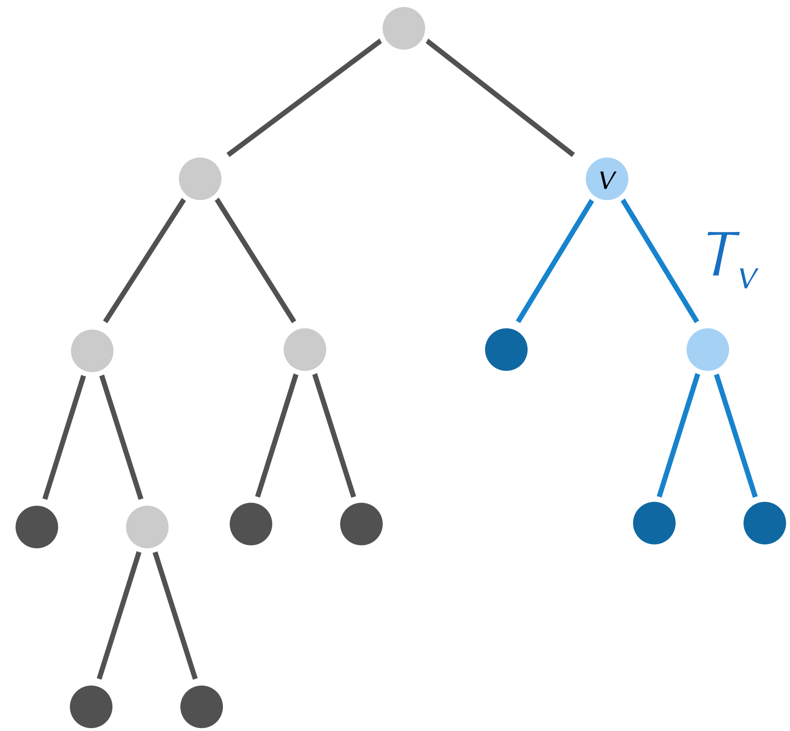


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Dynamic Programming Algorithm

Let T_v denote the subtree of T whose root is v.

Define $s_k(v)$ as the minimum parsimony score of T_v over all labelings of T_v, assuming that v is labeled by k.

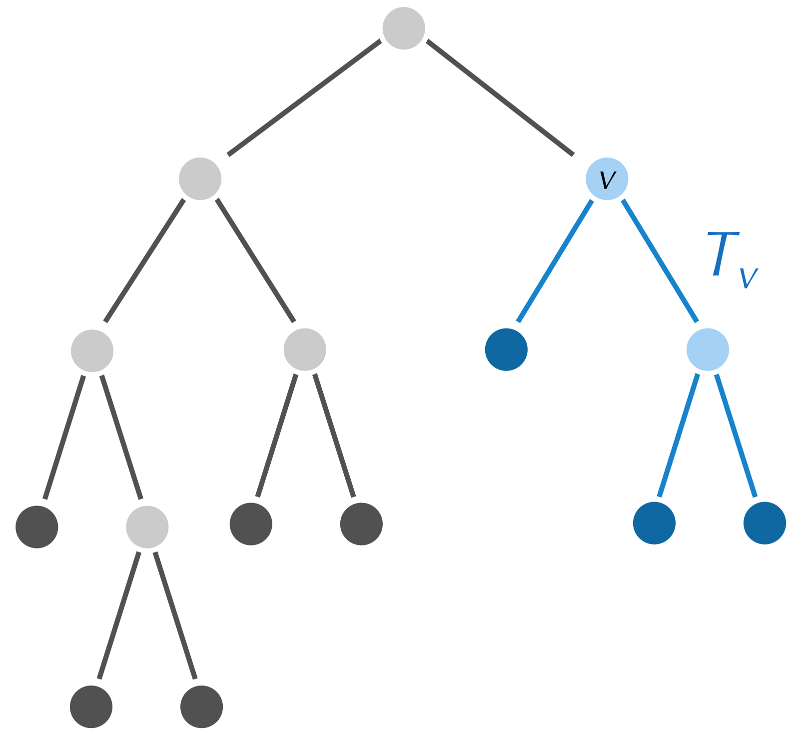


Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Dynamic Programming Algorithm

Let T_v denote the subtree of T whose root is v.

Define $s_k(v)$ as the minimum parsimony score of T_v over all labelings of T_v, assuming that v is labeled by k.



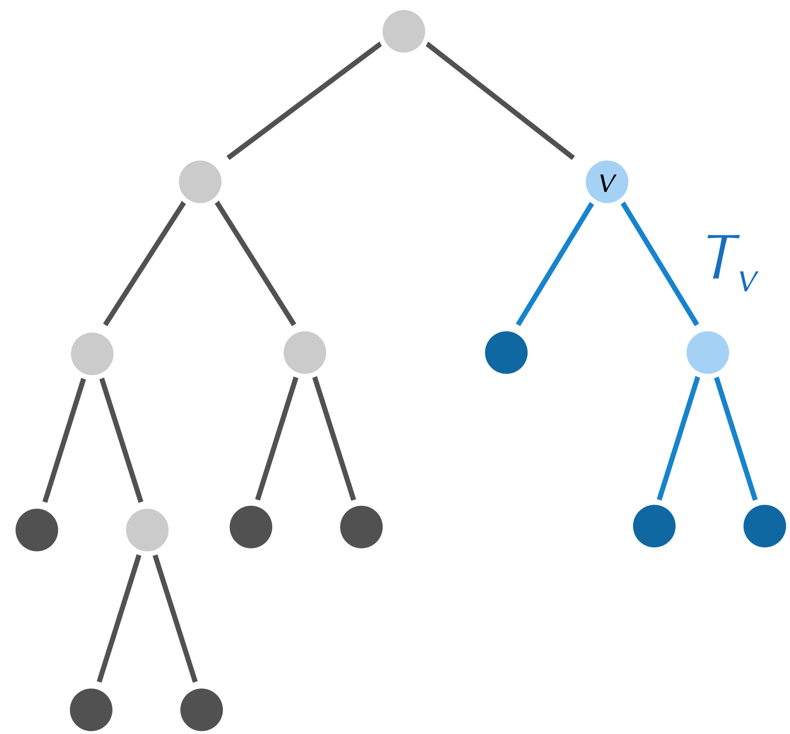
The minimum parsimony score for the tree is equal to the minimum value of $s_k(root)$ over all symbols k.

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Dynamic Programming Algorithm

For symbols i and j, define

- $\alpha_{i,j} = 0$ if $i = j$
- $\alpha_{i,j} = 1$ otherwise.

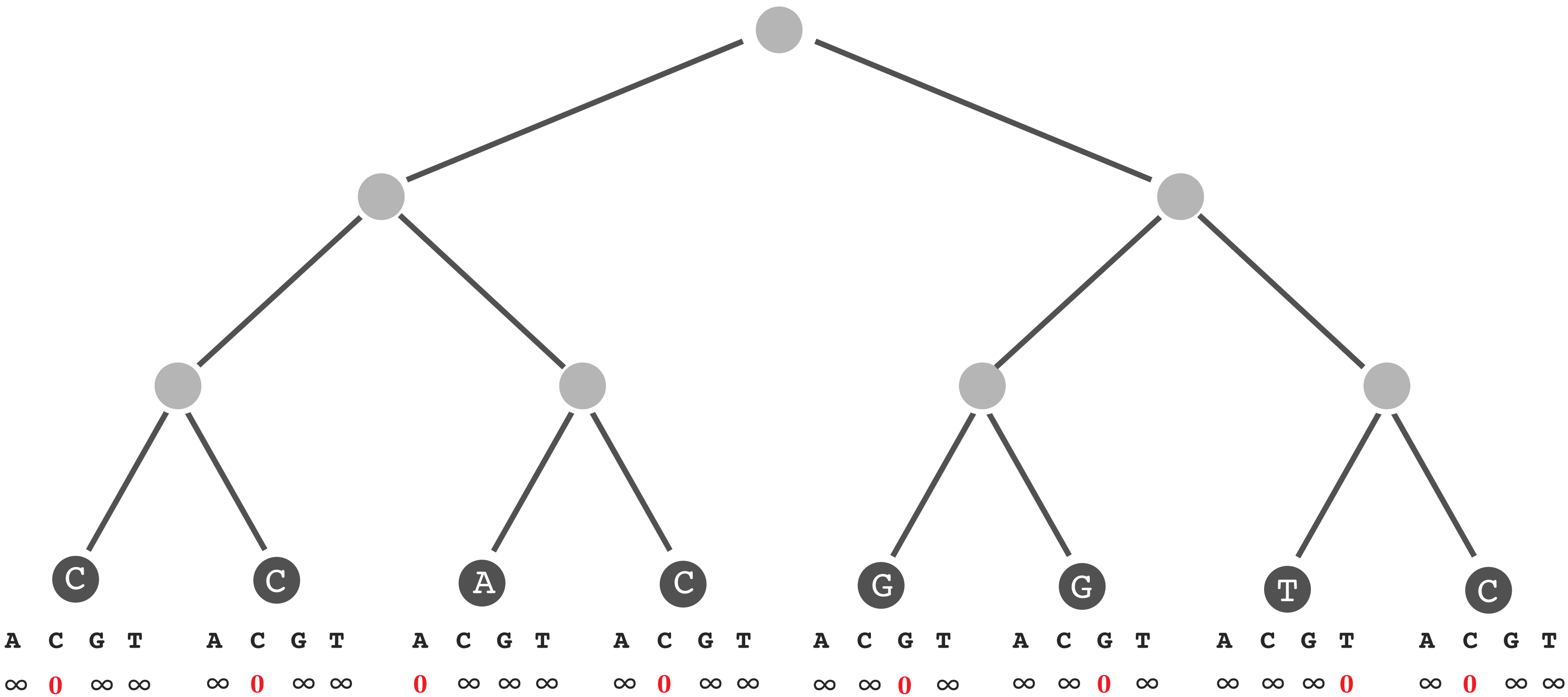


Theorem: The following recurrence relation holds:

$$s_k(v) = \min_{\text{all symbols } i} \{s_i(Daughter(v)) + \delta_{i,k}\} + \min_{\text{all symbols } i} \{s_i(Son(v)) + \delta_{j,k}\}$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

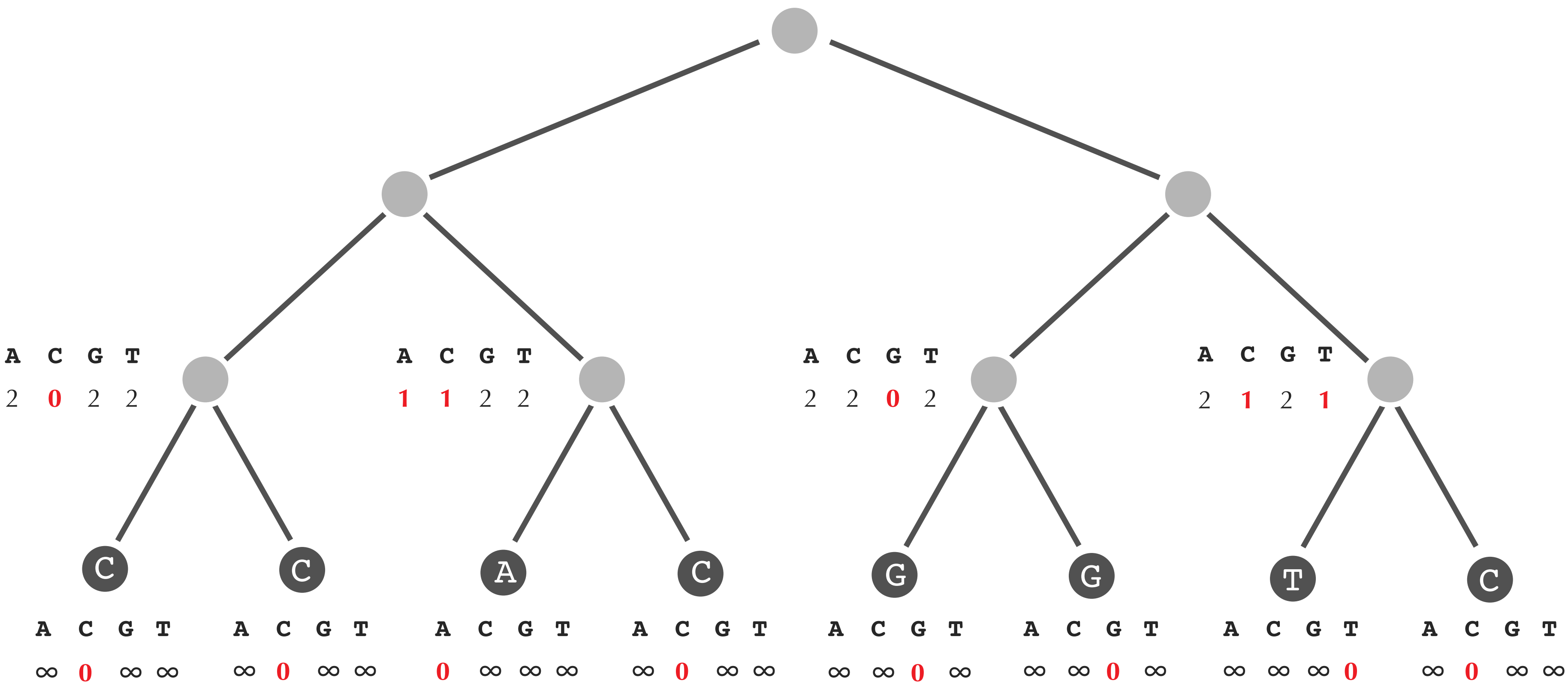
A Dynamic Programming Algorithm



$$s_k(v) = \min_{\text{all symbols } i} \{s_i(Daughter(v)) + \boldsymbol{\delta}_{i,k}\} + \min_{\text{all symbols } i} \{s_i(Son(v)) + \boldsymbol{\delta}_{j,k}\}$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

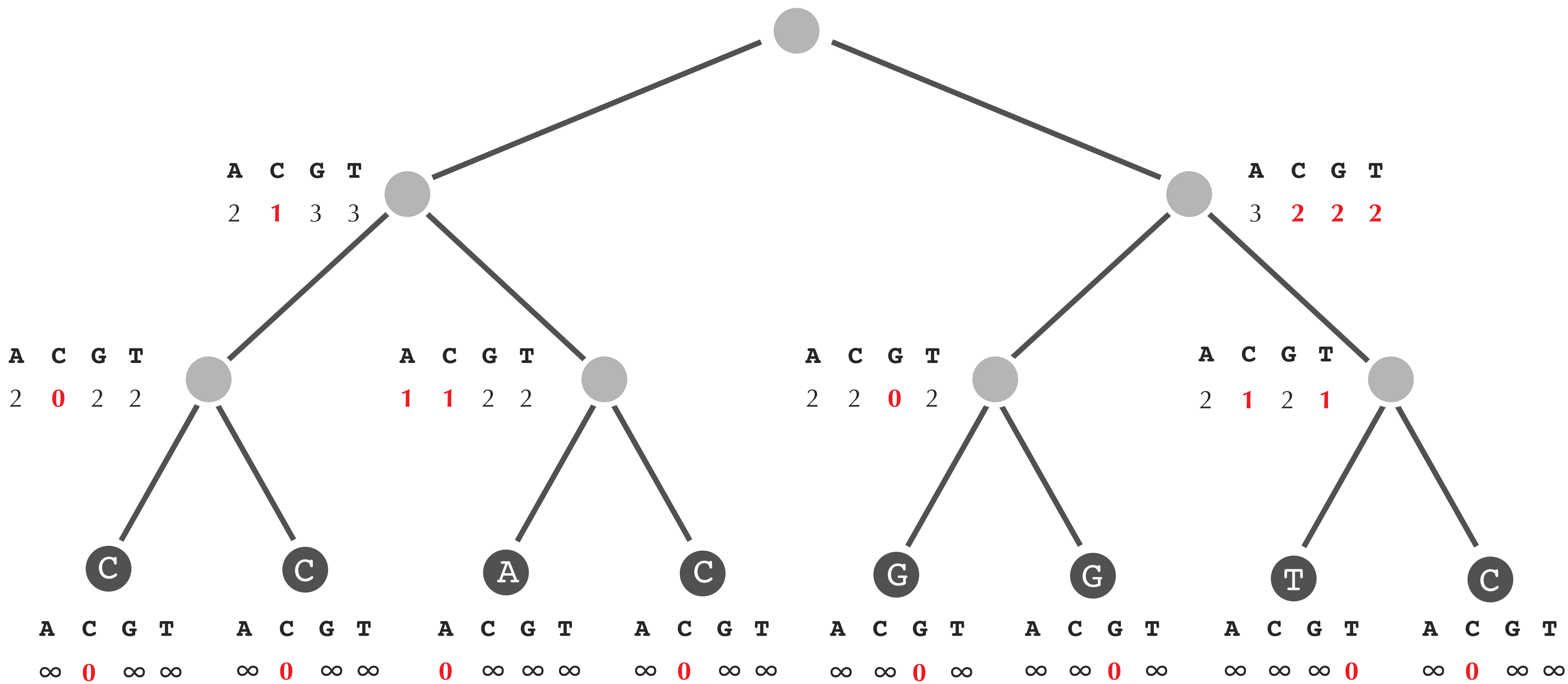
A Dynamic Programming Algorithm



$$s_k(v) = \min_{\text{all symbols } i} \{s_i(Daughter(v)) + \boldsymbol{\delta}_{i,k}\} + \min_{\text{all symbols } i} \{s_i(Son(v)) + \boldsymbol{\delta}_{j,k}\}$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

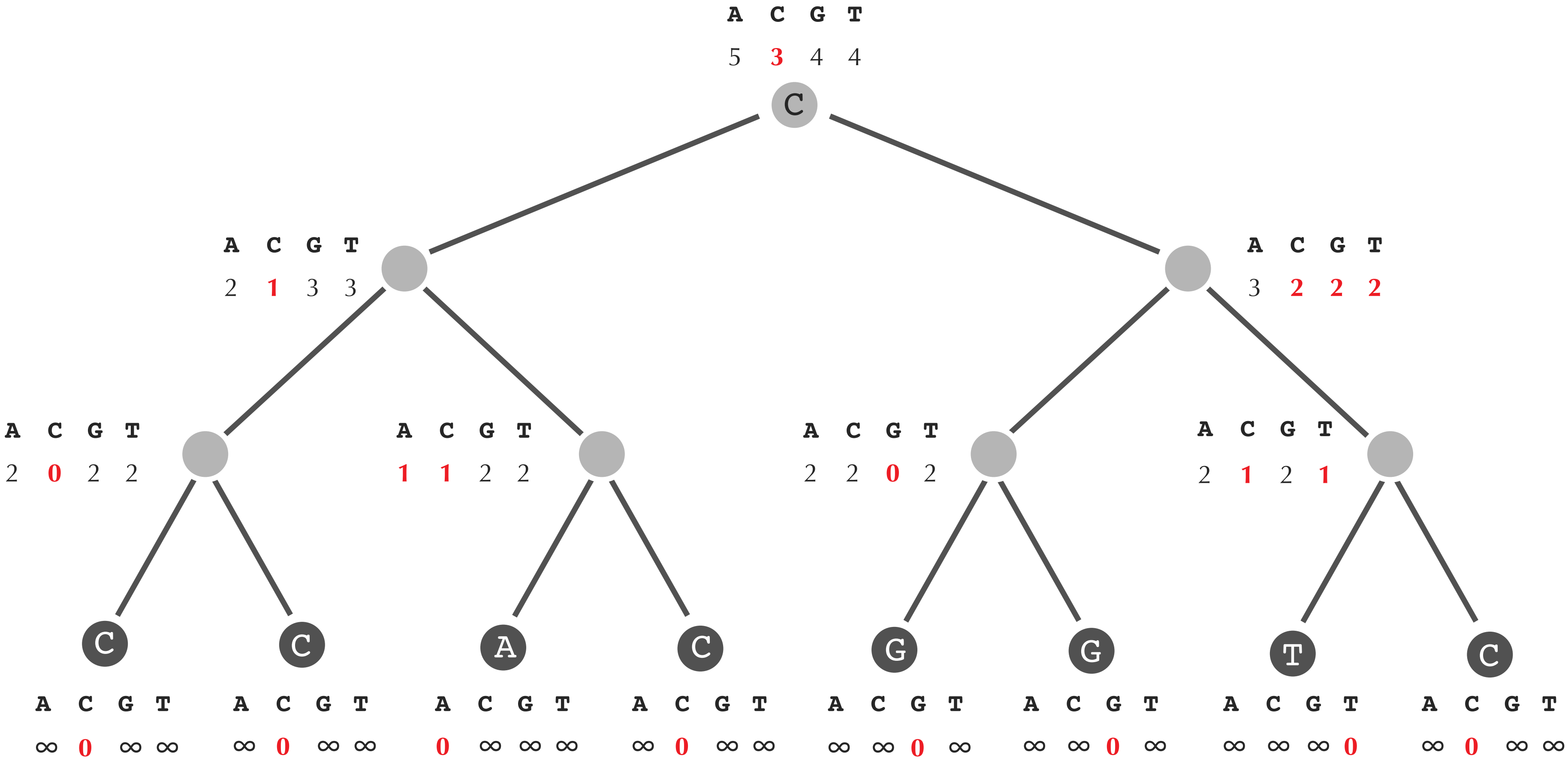
A Dynamic Programming Algorithm



$$s_k(v) = \min_{\text{all symbols } i} \{s_i(Daughter(v)) + \boldsymbol{\delta}_{i,k}\} + \min_{\text{all symbols } i} \{s_i(Son(v)) + \boldsymbol{\delta}_{j,k}\}$$

Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

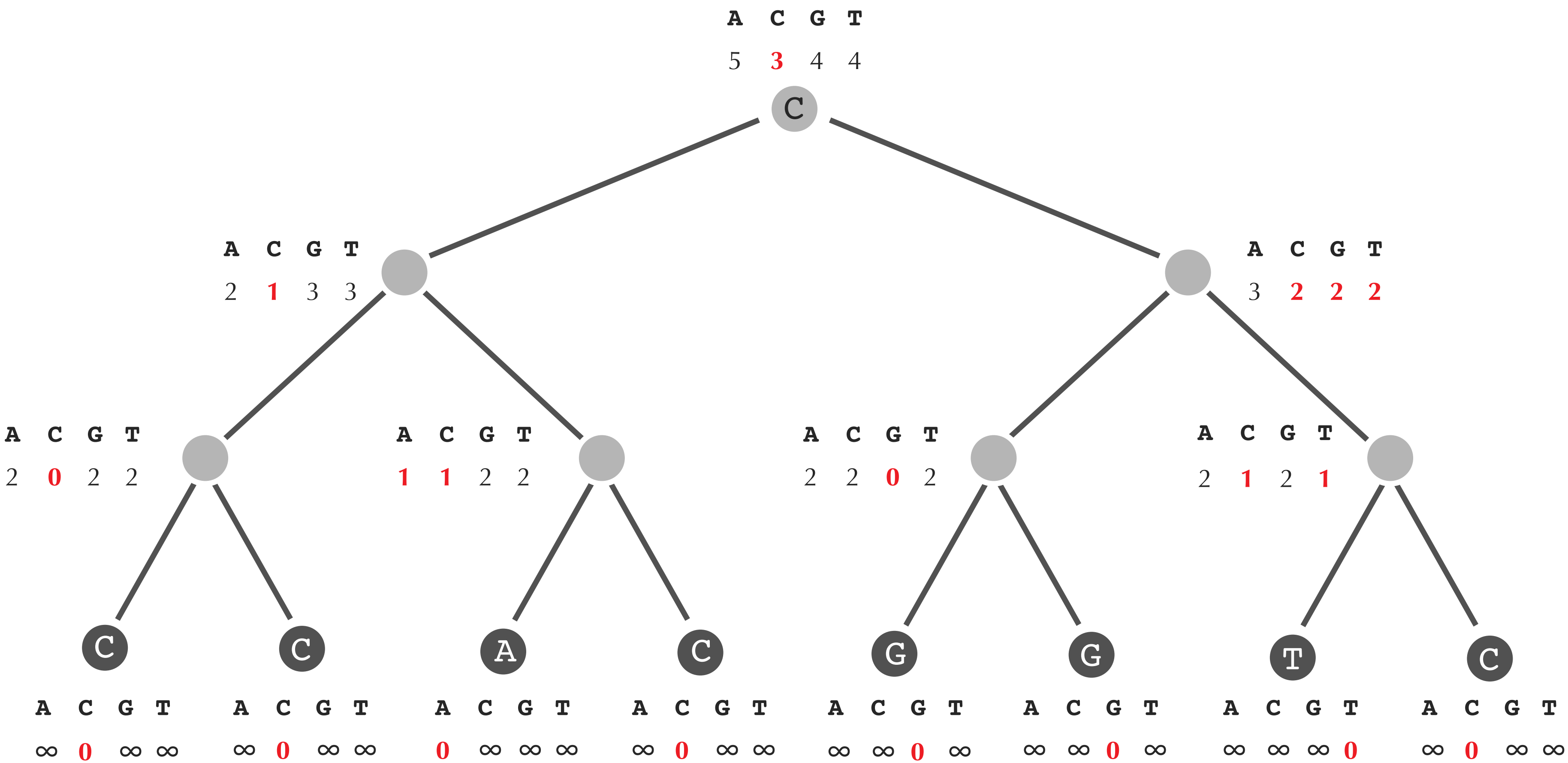
A Dynamic Programming Algorithm



$$s_k(v) = \min_{\text{all symbols } i} \{s_i(Daughter(v)) + \boldsymbol{\delta}_{i,k}\} + \min_{\text{all symbols } i} \{s_i(Son(v)) + \boldsymbol{\delta}_{j,k}\}$$

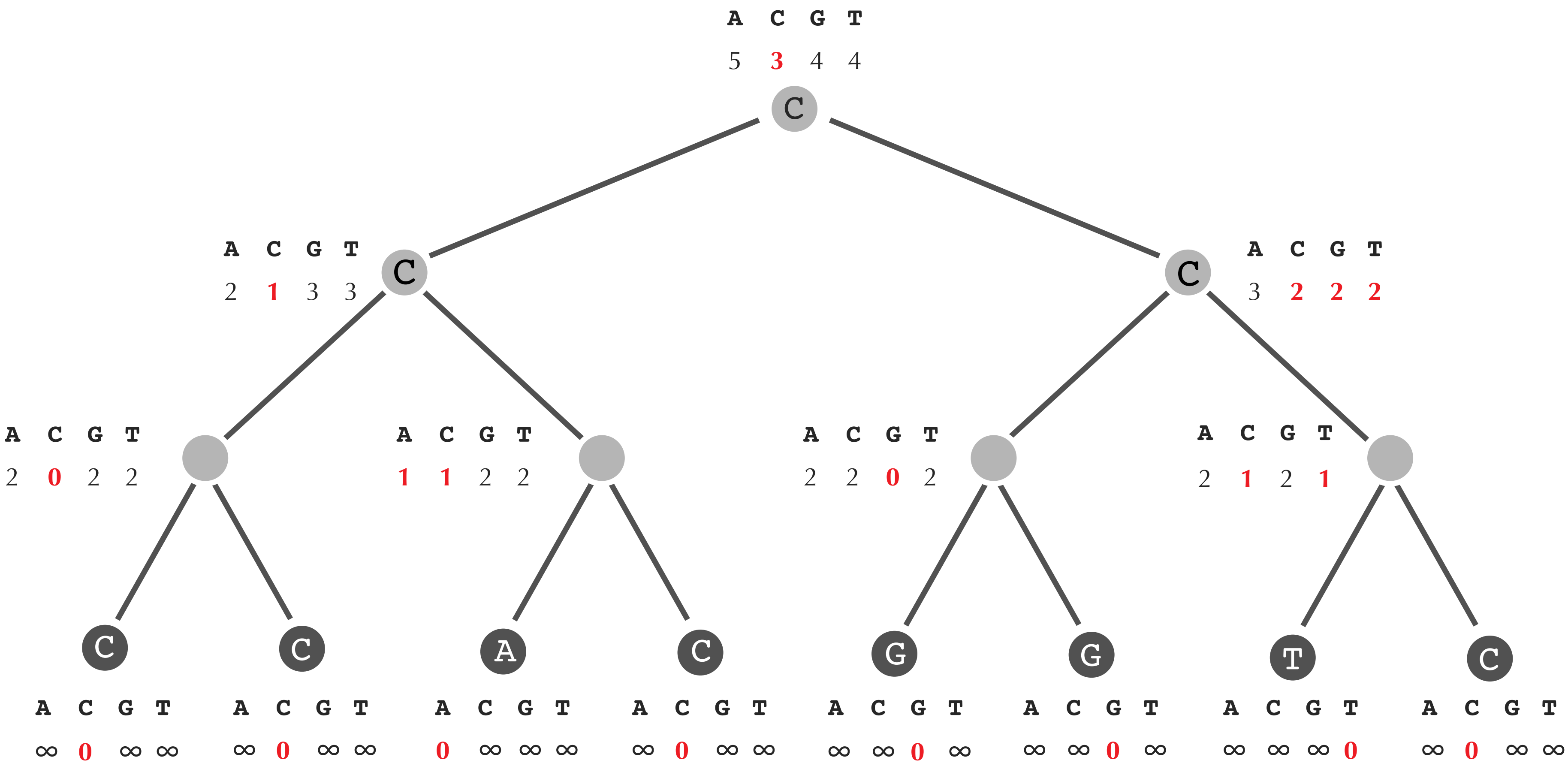
Bioinformatics Algorithms: An Active Learning Approach. © 2018 Compeau and Pevzner.

A Dynamic Programming Algorithm



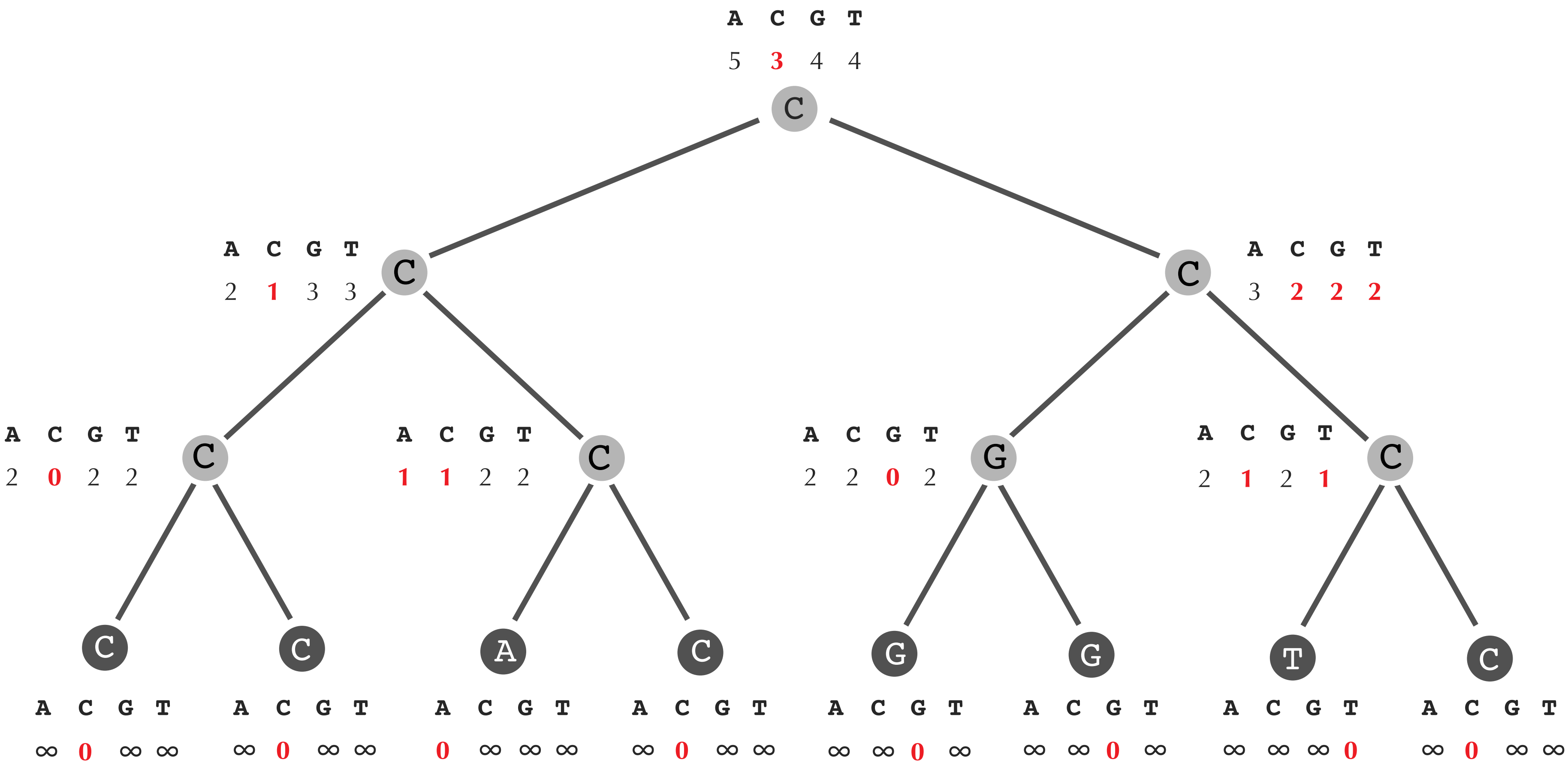
Checkpoint: How should we "backtrack" to fill in the remaining nodes of the tree?

A Dynamic Programming Algorithm



Checkpoint: How should we “backtrack” to fill in the remaining nodes of the tree?

A Dynamic Programming Algorithm



Checkpoint: How should we "backtrack" to fill in the remaining nodes of the tree?