1. THE FASTEST OUTBREAK

Trouble at the Metropole Hotel

On February 21, 2003, a Chinese doctor named Liu Jianlun flew to Hong Kong to attend a wedding and checked into Room 911 of the Metropole Hotel. The next day, he became too ill to attend the wedding and was admitted to a hospital. Two weeks later, Dr. Liu was dead.

On his deathbed, Liu told doctors that he had recently treated sick patients in Guangdong Province, China where a deadly, highly contagious respiratory illness had infected hundreds of people. The Chinese government had made brief mention of this incident to the World Health Organization but had concluded that the likely culprit was a common bacterial infection. By the time anyone realized the severity of the disease, it was already too late to stop the outbreak. On February 23, a man who had stayed across the hall from Dr. Liu at the Metropole traveled to Hanoi and died after infecting 80 people. On February 26, a woman checked out of the Metropole, traveled back to Toronto, and died after initiating an outbreak there. On March 1, a third guest was admitted to a hospital in Singapore, where sixteen additional cases of the illness arose within two weeks.

Consider that it took four years for the Black Death, which killed over a third of all Europeans in the 14th Century, to travel from Constantinople to Kiev. Or that HIV took two decades to circle the globe. In contrast, this mysterious new disease had crossed the Pacific Ocean within a week of entering Hong Kong.

As health officials braced for the impact of the fastest-traveling pandemic in human history, panic set in. Businesses were closed, sick passengers were removed from airplanes, and Chinese officials threatened to execute infected patients who violated quarantine.

International travel may have helped the disease spread rapidly, but international collaboration would eventually contain it. In a matter of a few weeks, biologists identified a virus that had caused the epidemic and sequenced its genome. In the process, the mysterious new disease earned a name: **Severe Acute Respiratory Syndrome**, or **SARS**.

The evolution of SARS

The virus causing SARS belongs to a family of viruses called **coronaviruses**, which are named after the Latin *corona* (meaning “crown”) because the virus particle resembles the sun’s corona

(see the figure below). Coronaviruses infect the respiratory tracts of mammals and birds and typically cause only minor problems, like the common cold. Before SARS, no one believed that a coronavirus could wreak such havoc.

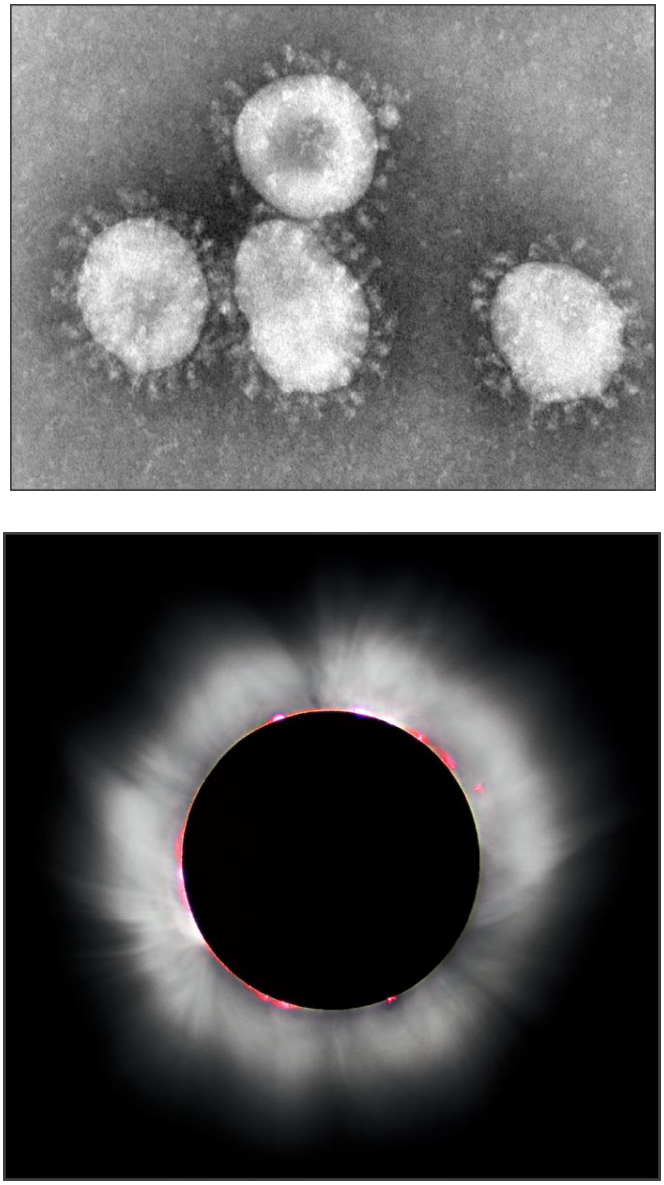


Figure: (Top) Coronavirus particles. (Bottom) A solar eclipse with the sun's corona visible. Photo: Luc Viatour.

Coronaviruses, influenza viruses, and HIV are all **RNA viruses**, meaning that they possess RNA instead of DNA. RNA replication has a higher error rate than DNA replication, and so RNA viruses are capable of mutating more quickly into divergent strains. The rapid mutation of RNA viruses explains why the flu shot changes from year to year and why there are many different subtypes of HIV.

SARS researchers initially hypothesized that, like HIV and influenza, the **SARS coronavirus** (abbreviated as **SARS-CoV**) had jumped from animals to humans. They first blamed birds as the likely suspect because of the similarities between the SARS outbreak and "bird flu", a form of influenza originating in chickens that is difficult to transmit to humans but is even deadlier than SARS, killing over half the people it infects. Yet when researchers sequenced the 29,751 nucleotide-long SARS-CoV genome in April 2003, it became evident that SARS did not come from birds because its genome did not resemble avian coronaviruses.

By fall 2003, researchers had sequenced many SARS-CoV strains from patients in various countries, but many questions still remained unanswered. How did SARS-CoV cross the species barrier to humans? When and where did it happen? How did SARS spread around the world, and who infected whom?

Each of these questions about SARS is ultimately related to the problem of constructing **evolutionary trees** (also known as **phylogenies**). For another example, by constructing an evolutionary tree of primate viruses related to HIV (figure below), scientists inferred that HIV was transmitted to humans on five separate occasions. But what algorithm did they use to construct this phylogeny?

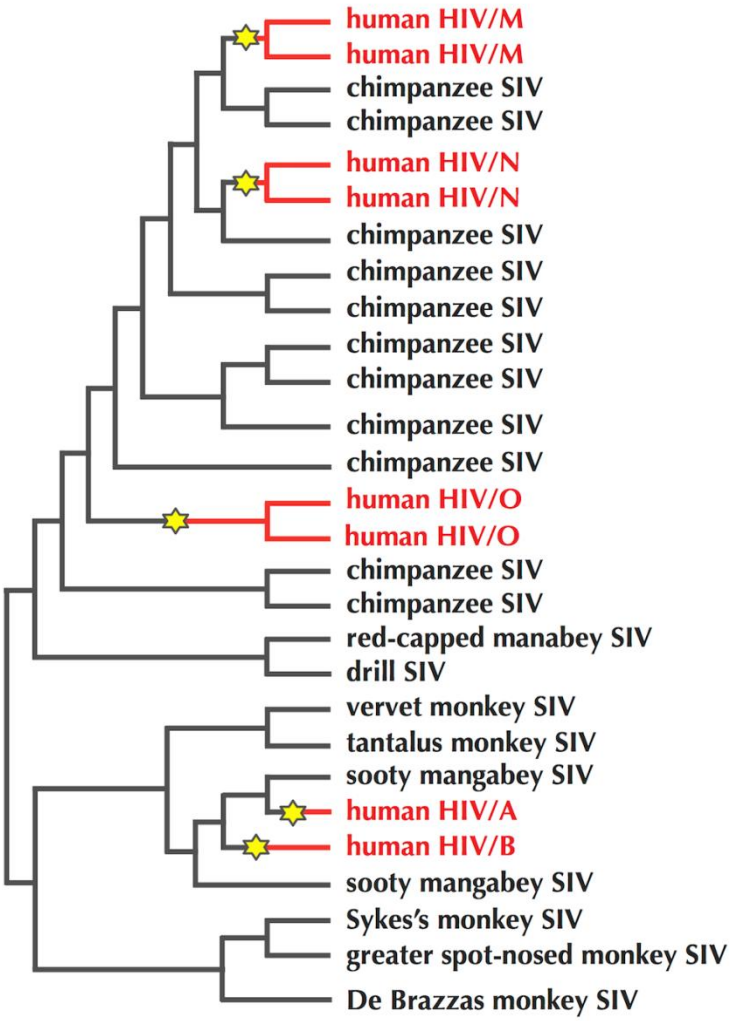


Figure: HIV comprises five different viral families, denoted as A, B, M, N, and O, with the M family responsible for 95% of all HIV infections. The five families are different offshoots of the evolutionary tree for Simian Immunodeficiency Virus (SIV), which infects primates. Stars indicate viruses transitioning from primates to humans. The A and B families originated in sooty mangabey monkeys, whereas the M, N, and O families originated in chimpanzees.

2. TRANSFORMING DISTANCE MATRICES INTO EVOLUTIONARY TREES

Constructing a distance matrix from coronavirus genomes

To determine how SARS jumped from animals to humans, scientists started sequencing coronaviruses from various species in order to determine which one is the most similar to SARS-CoV. However, constructing a multiple alignment of entire viral genomes is tricky because viral genes are often rearranged, inserted, and deleted. For this reason, scientists focused on only one of the six genes in SARS-CoV. This gene encodes the **Spike protein**, which identifies and binds to receptor sites on the host's cell membrane.

In SARS-CoV, the Spike protein is 1,255 amino acids long and has rather weak similarity with Spike proteins in other coronaviruses. However, even these subtle similarities turned out to be sufficient for constructing a multiple alignment of Spike proteins across various coronaviruses.

After constructing a multiple alignment of genes from n different species, biologists often transform this alignment into an $n \times n$ **distance matrix** D. In many cases, $D_{i,j}$ represents the number of differing symbols between the genes representing rows i and j of the alignment (see the figure below). However, distance matrices can be constructed using a variety of different distance functions in order to suit different applications. For example, $D_{i,j}$ could also represent the edit distance between genes from the i-th and j-th species. Or, a distance matrix for n genomes could be constructed from the 2-break distances between each pair of genomes.

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

Figure: A multiple alignment of hypothetical DNA sequences from four species, along with the distance matrix produced by counting the number of differing symbols between each pair of rows in this multiple alignment.

Regardless of which distance function we use, in order to be a distance matrix, D must satisfy three properties. It must be **symmetric** (for all i and j, $D_{i,j} = D_{j,i}$), **non-negative** (for all i and j, $D_{i,j} \geq 0$) and satisfy the **triangle inequality** (for all i, j, and k, $D_{i,j} + D_{j,k} \geq D_{i,k}$).

Exercise 1: Prove that if $D_{i,j}$ is equal to the number of differing symbols between rows i and j in a multiple alignment, then D is symmetric, non-negative, and satisfies the triangle inequality.

By the end of 2003, bioinformaticians had sequenced many coronaviruses taken from a variety of animals and SARS patients and then computed the associated distance matrix. They needed to use this information in order to construct a coronavirus phylogeny and understand the origin and spread of the SARS epidemic.

Evolutionary trees as graphs

You may have noticed that the HIV tree in the Figure on step 2 has the structure of a graph. Furthermore, the figure below shows the representation of the phylogeny of all life as a graph.

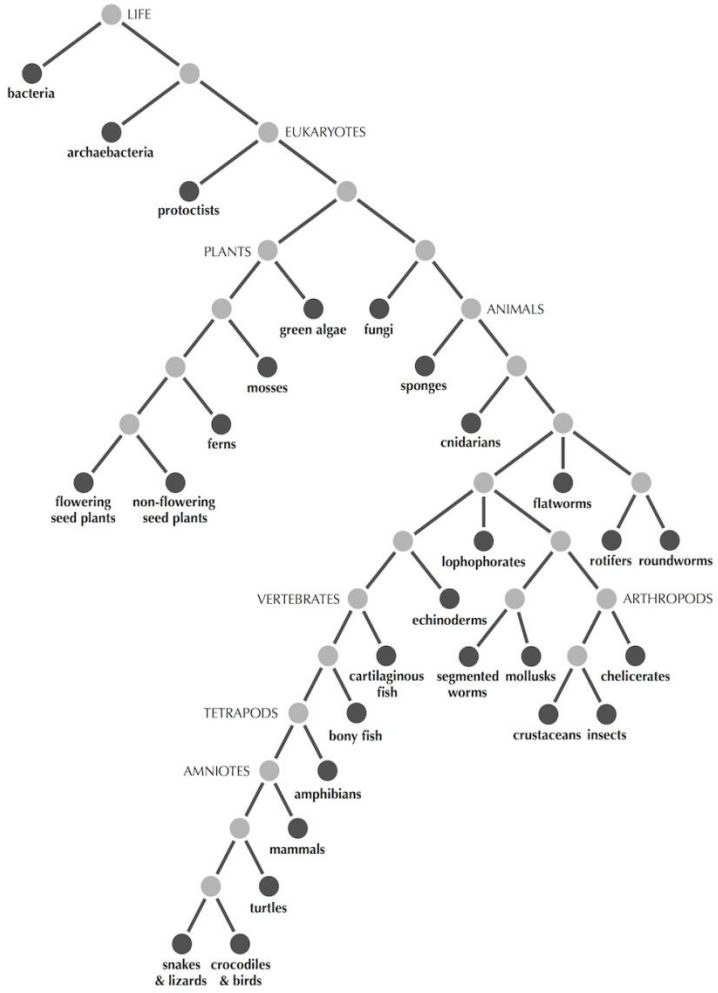


Figure: A connected graph without cycles that models an evolutionary tree of life on Earth. Present-day species are shown as darker nodes.

Graphs used to model phylogenies share two properties. They are connected (i.e. it is possible to reach any node from any other node), and they contain no cycles. For this reason, we will define a **tree** as a connected graph without cycles.

See the figure below for a few additional examples; the darker nodes are **leaves**, or nodes having degree 1 (the degree of a node is the number of edges connected to that node). Nodes of larger degree are called **internal nodes**.

Exercise 2: Prove the following statements:

- Every tree with at least two nodes has at least two leaves.
- Every tree with n nodes has n - 1 edges;
- There exists exactly one path connecting every pair of nodes in a tree. Hint: what would happen if there were two different paths connecting a pair of nodes? What would happen if there were no paths connecting a pair of nodes?

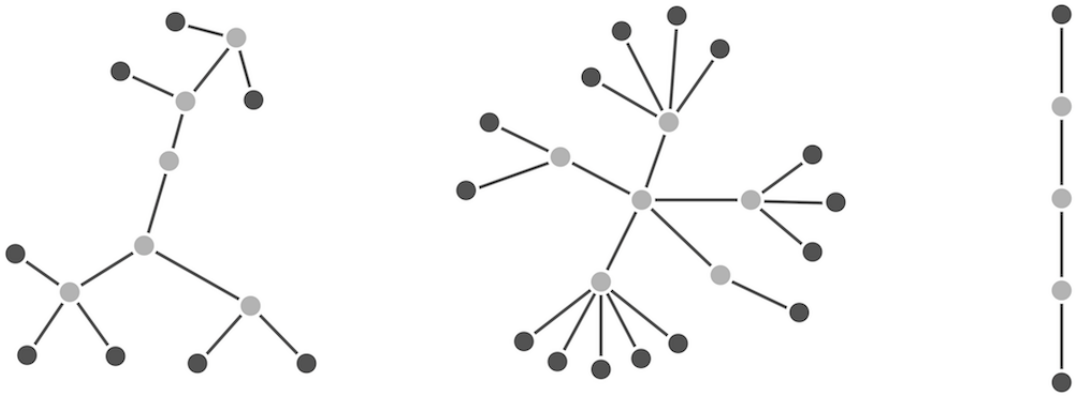


Figure: Trees come in a variety of different shapes. In each of the three trees shown, leaves have been drawn darker than internal nodes.

Take another look at the tree of life figure. You will see that present-day species have been assigned to the leaves of this tree. Internal nodes represent unknown ancestor species.

Given a leaf j, there is only one node connected to j by an edge, which we call the **parent** of j, denoted *Parent*(j). An edge connecting a leaf to its parent is called a **limb**.

In a **rooted tree**, one node is designated as a special node called the **root**, and the edges in the tree automatically inherit an implicit orientation away from the root, which is placed at the top or left of the tree (see the figure below). This edge orientation models time: the ancestor of all species in the tree is found at the root, and evolution proceeds from the root outward through the tree. Trees without a designated root are called **unrooted**.

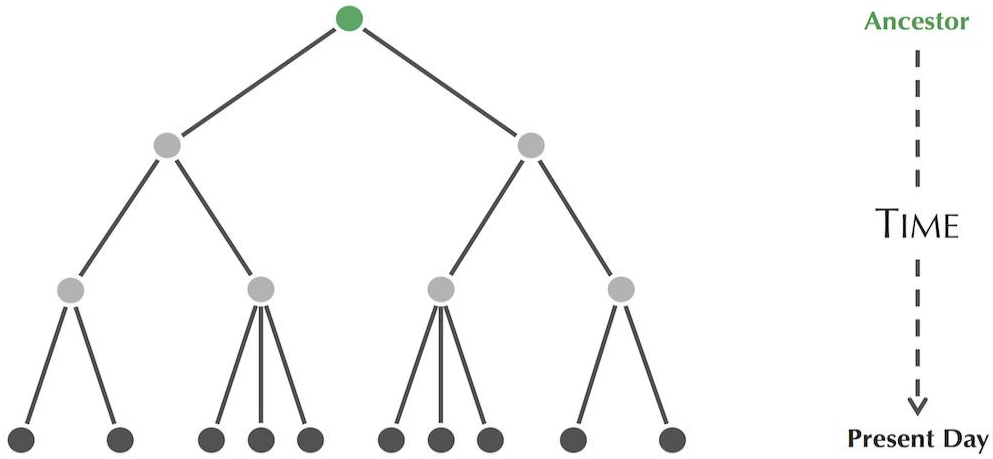
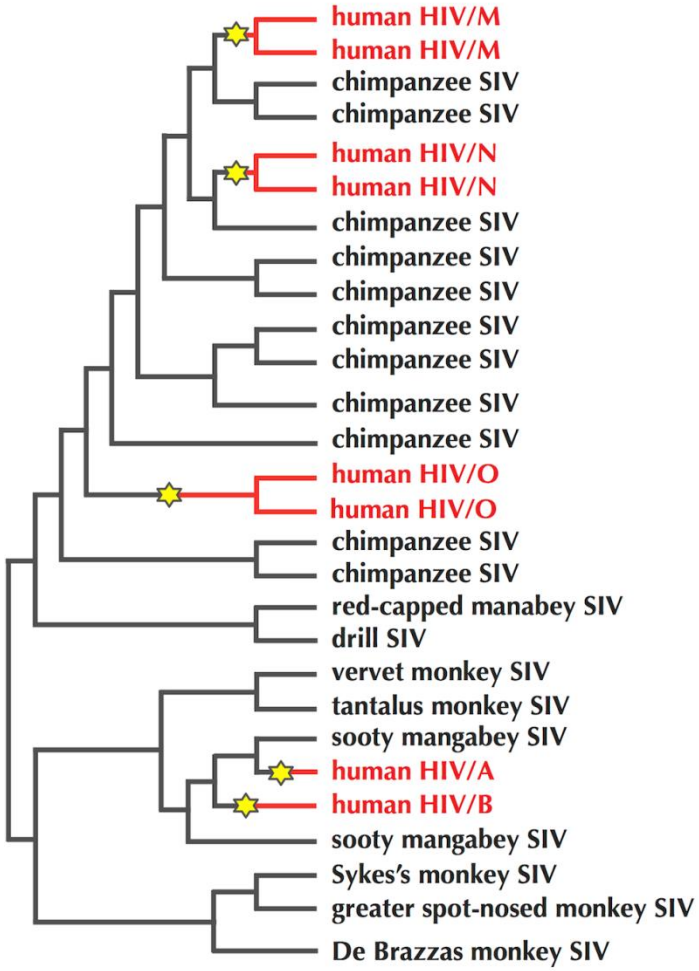


Figure: A rooted tree, with the root (representing an ancestor of all species in the tree) indicated in green at the top of the tree. The presence of the root implies an orientation of edges in the tree away from the root.

Question 1: The HIV phylogeny we encountered before is reproduced below. Where would you place the root in this phylogeny?



Here, we will analyze rooted trees when we will attempt to infer the node corresponding to the ancestor of all species in the tree; otherwise, we will analyze unrooted trees. The figure below

shows an unrooted tree of HIV viruses produced from a different dataset than the one used to create the figure in the previous step. By proposing two additional subtypes of HIV, it illustrates that the classification of HIV into five families is not written in stone.

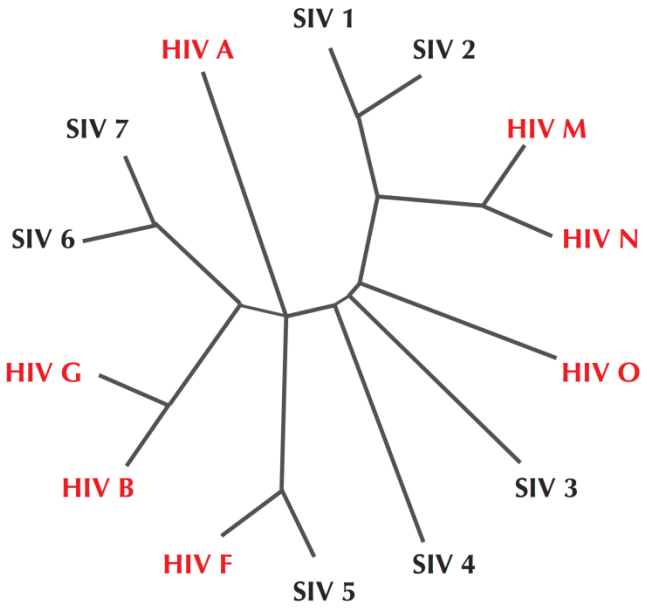


Figure: An unrooted tree of HIV and SIV viruses that suggests additional viral families F and G in addition to the viral families A, B, M, N, and O shown in the figure on the previous step.

Distance-based phylogeny construction

We will first focus on deriving an unrooted tree from a distance matrix. The leaves of this tree should correspond to the species represented by the matrix (with internal nodes corresponding to unknown ancestral species). To reflect the evolutionary distance between species in a tree, we assign each edge a non-negative length representing the distance between the organisms that the edge connects, as shown in the figure below.

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

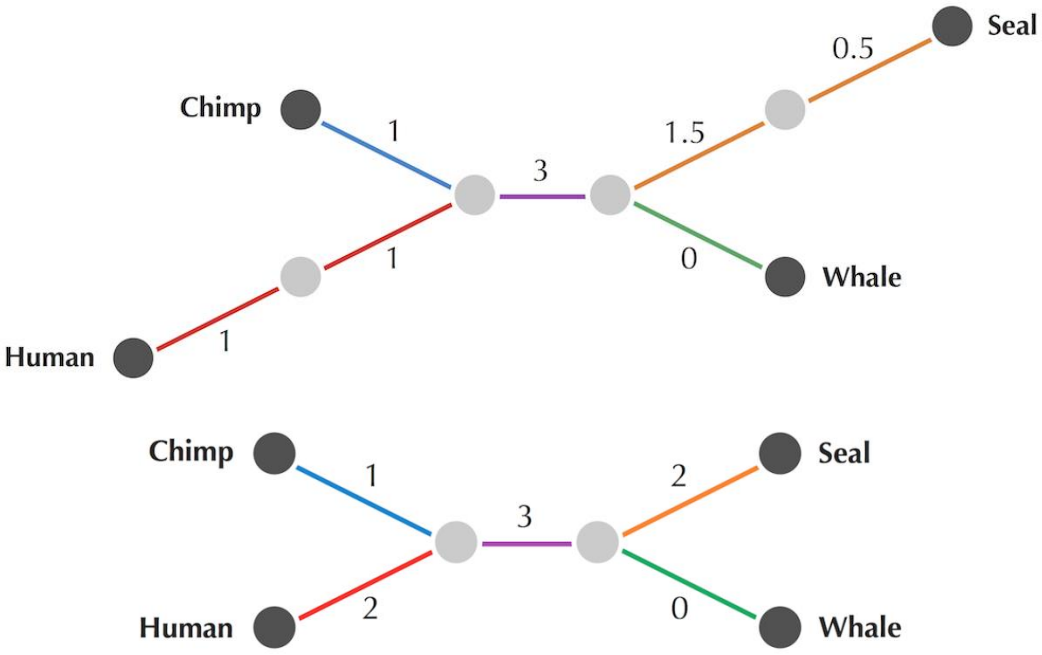


Figure: The toy distance matrix constructed from a multiple alignment as well as two unrooted trees fitting this distance matrix.

In this section, we define the length of a path in a tree as the sum of the lengths of its edges (rather than the number of edges on the path). As a result, the evolutionary distance between two present-day species corresponding to leaves i and j in a tree T is equal to the length of the unique path connecting i and j, denoted $d_{i,j}(T)$.

Distance-Based Phylogeny Problem: *Reconstruct an evolutionary tree fitting a distance matrix.*

Input: A distance matrix.

Output: A tree fitting this distance matrix.

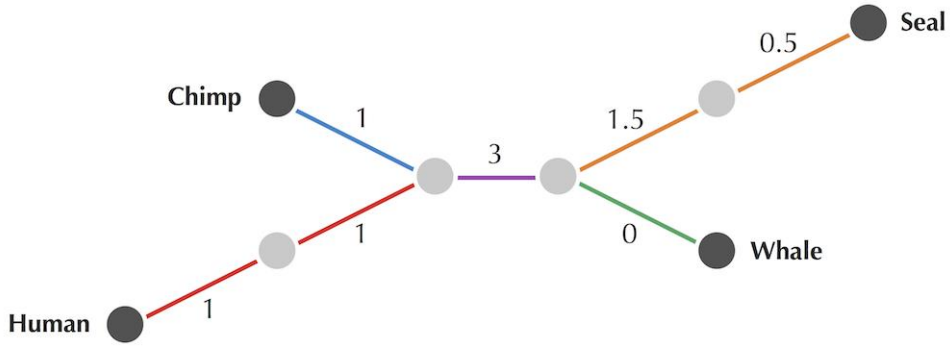
Question 2: Does the Distance-Based Phylogeny Problem always have a solution? Argument your answer.

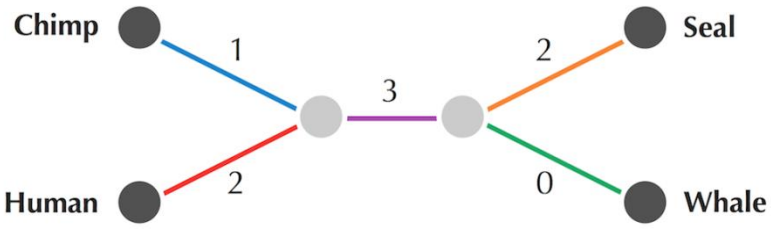
Not every distance matrix has a tree fitting it. We therefore call a distance matrix **additive** if there exists a tree that fits this matrix and **non-additive** otherwise (an example 4 x 4 non-additive matrix is shown below). The term "additive" is used because the lengths of all edges along the path between leaves i and j in a tree fitting a matrix D add to $D_{i,j}$.

	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

Both trees below fit the toy distance matrix, so it would be nice to have a notion of a "canonical" tree fitting a distance matrix. We say that a path in a tree is **non-branching** if every node other than the beginning and ending node of the path has degree equal to 2. A non-branching path is **maximal** if it is not a subpath of an even longer non-branching path. If we substitute every maximal non-branching path by a single edge whose length is equal to the length of the path, then the tree in figure on top becomes the tree in figure on the bottom. In general, after such a transformation, there are no nodes of degree 2; a tree satisfying this property is called a **simple tree**.

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





It turns out that if a matrix is additive, then there exists a *unique* simple tree fitting this matrix. In the Distance-Based Phylogeny Problem, we will therefore use the terminology *Tree*(*D*) to denote the simple tree fitting the additive distance matrix *D*. Our question, then, is how to construct *Tree*(*D*) from *D*.

Exercise 3: Prove that every simple tree with n leaves has at most n - 2 internal nodes.

3. TOWARD AN ALGORITHM FOR DISTANCE MATRIX

A quest for neighboring leaves

A natural first step for solving the Distance-Based Phylogeny Problem would be to ensure that the two closest species with respect to the distance matrix D correspond to neighbors in *Tree*(D). In other words, the minimum value $D_{i,j}$ should correspond to leaves i and j having the same parent. In the rest of this section, when we refer to the minimum element of a matrix, we are referring to a minimum **off-diagonal** element, i.e., a value $D_{i,j}$ such that $i \neq j$.

Theorem: *Every simple tree with at least three nodes has a pair of neighboring leaves.*
Proof: Given a simple tree T with at least three nodes, consider a path $P = (v_1, \ldots, v_k)$ that has the maximum number of nodes of any path in T. Because T has at least three nodes, k must be at least 3. Furthermore, nodes v_1 and v_k must be leaves, since otherwise we could extend P into a longer path. Because T is simple, each internal node of T has degree at least 3. Thus, node v_2, which is the parent of v_1, must have at least three adjacent nodes: v_1, v_3, and yet another node w. We claim that w is a leaf, which would imply that leaves v_1 and w are neighbors. We will proceed by contradiction: if w were not a leaf, then since T is simple, w would be adjacent to another node u. As a result, we could form the path $P' = (u, w, v_2, v_3, \ldots, v_k)$, which contains $k + 1$ nodes and contradicts our original assumption that P has the maximum number of nodes. Thus, w must be a leaf, implying that v_1 and w are neighbors and establishing the result.

The figure below illustrates that for neighboring leaves i and j sharing a parent node m, the following equality holds for every other leaf k in the tree:

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

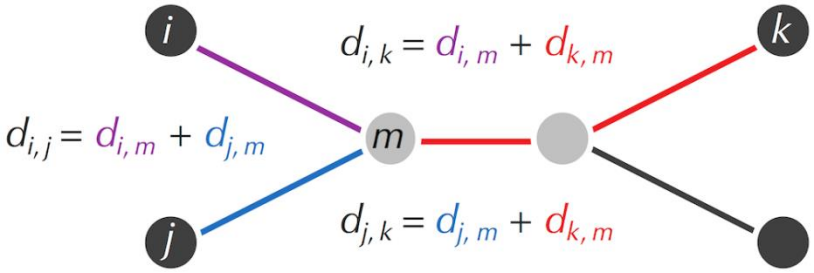


Figure: For neighboring leaves i and j and their parent node m, $d_{k,m}=(d_{i,k}+d_{j,k}-d_{i,j})/2$ for every other leaf k in the tree**.**

Since i, j, and k are leaves, we can compute the distance $d_{k,m}$ between nodes k and m in terms of elements of the additive distance matrix D:

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

In the case when the parent m has degree 3 (as in the figure on the previous step), removing leaves i and j from the tree turns m into a leaf and thus reduces the total number of leaves (see figure below). This operation is equivalent to removing rows i and j as well as columns i and j from D, then adding a new row and column corresponding to their parent m, where the distances from m to other leaves are computed according to the above formula.

Exercise 4: We have just described how to reduce the size of the tree as well as the dimension of the distance matrix D if the parent node (m) has degree 3. Design a similar approach in the case that the degree of m is larger than 3.

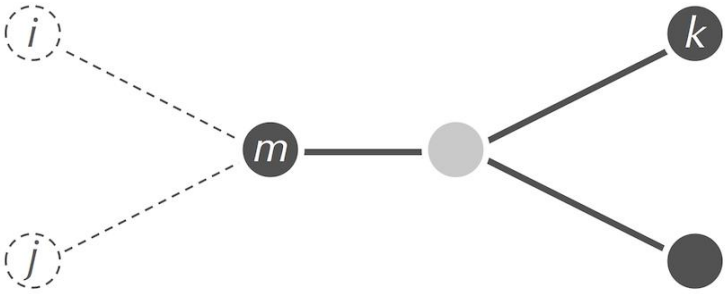


Figure: Removing leaves i and j from the tree turns m into a leaf (we assume that m has degree 3). The distances from this new leaf to any other leaf k can be recomputed as $d_{k,m}=(D_{i,k}+D_{j,k}-D_{i,j})/2$

This discussion implies a recursive algorithm for the Distance-Based Phylogeny Problem:

- find a pair of neighboring leaves i and j by selecting the minimum element $D_{i,j}$ in the distance matrix;

- replace *i* and *j* with their parent, and recompute the distances from this parent to all other leaves as described above;
- solve the Distance-Based Phylogeny problem for the smaller tree;
- add the previously removed leaves *i* and *j* back to the tree.

Exercise 5: Apply this recursive approach to the additive distance matrix shown in the figure below. (Solve this exercise by hand.) Argument your results.

	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

Computing limb lengths

If you attempted the preceding exercise, then you were likely driven crazy. The reason why is that in the first step of our proposed algorithm, we assumed that a minimum element of an additive distance matrix corresponds to neighboring leaves. Yet as illustrated in figure below, this assumption is not necessarily true! Thus, we need a new approach to the Distance-Based Phylogeny Problem, as finding the animal coronavirus that is the smallest distance from SARS-CoV may not be the best way to identify the animal reservoir of SARS.

	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

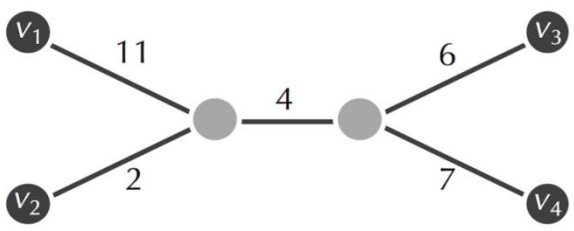


Figure: The distance matrix from the previous step along with the simple tree fitting this distance matrix. The two closest leaves in this tree (*j* and *k*) are not neighbors.

Our proposed recursive approach may have failed, but using recursion was a good idea, and so we will explore a different recursive algorithm. Rather than looking for a *pair* of neighbors in *Tree*(*D*), we will instead reduce the size of the tree by trimming its leaves *one at a time*. Of course, we don't know *Tree*(*D*), and so we must somehow trim leaves in *Tree*(*D*) by analyzing the distance matrix.

As a first step toward constructing *Tree(D)*, we will address the more modest goal of computing the lengths of limbs in *Tree(D)*. So, given a leaf *j* in a tree, we denote the length of the limb connecting *j* with its parent as *LimbLength*(*j*). Edges that are not limbs must connect two internal nodes and are therefore called **internal edges**.

Limb Length Problem: *Compute the length of a limb in a tree defined by an additive distance matrix.*

Input: An additive distance matrix *D* and an integer *j*.

Output: *LimbLength*(*j*), the length of the limb connecting leaf *j* to its parent in *Tree*(*D*).

To compute *LimbLength*(*j*) for a given leaf *j*, note that because *Tree*(*D*) is simple, we know that *Parent*(*j*) has degree at least 3 (unless *Tree*(*D*) has only two nodes). We can therefore think of *Parent*(*j*) as dividing the other nodes of *Tree*(*D*) into at least three **subtrees**, or smaller trees that would remain if we were to remove *Parent*(*j*) along with any edges connecting it to other nodes (see figure below). Because *j* is a leaf, it must belong to a subtree by itself; we call this subtree T_j. This brings us to the following result.

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

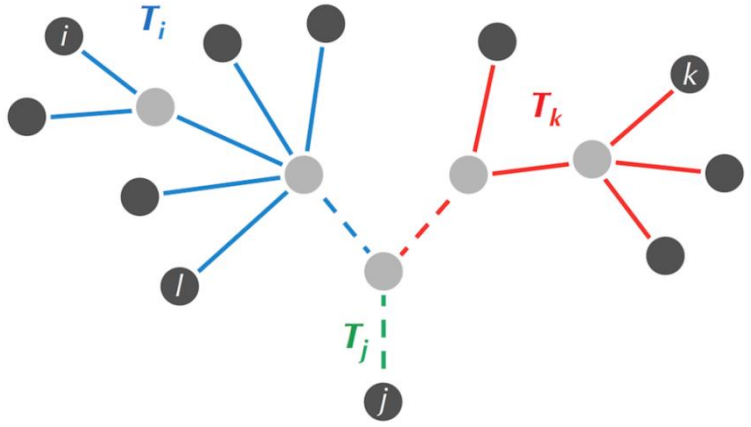


Figure: A simple tree with selected leaves *i*, *j*, *k*, and *l*. Removing the parent of *j* (along with the three dashed edges connecting it to other nodes) would separate this tree into three subtrees, whose edges are shown in different colors.

Leaves i and l belong to T_i, whereas leaf k belongs to T_k. Leaf j belongs to T_j, which contains a single node.

Proof of Limb Length Theorem: A given pair of leaves can belong to the same subtree or to different subtrees in the figure on the previous step, reproduced above. So first assume that leaves i and k belong to different subtrees T_i and T_k. Because $Parent(j)$ is on the path connecting i to k, it follows that

$$d_{i,j}=d_{i,Parent(j)}+LimbLength(j) \text{ and}$$

$$d_{j,k}=d_{k,Parent(j)}+LimbLength(j).$$

Adding these two equations yields

$$d_{i,j}+d_{j,k}=d_{i,Parent(j)}+d_{k,Parent(j)}+2\cdot LimbLength(j)$$

Because $d_{i,Parent(j)}+d_{k,Parent(j)}$ is equal to $d_{i,k}$ it follows that

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

On the other hand, assume that leaves i and l belong to the same subtree (see the figure on the previous step). Then the path from i to l does not pass through $Parent(j)$, and so we have the *inequality*

$$d_{i,Parent(j)}+d_{l,Parent(j)}\geq d_{i,l}$$

Combining this with the equation

$$d_{i,j}+d_{j,l}=d_{i,Parent(j)}+d_{l,Parent(j)}+2\cdot LimbLength(j)$$

yields that

$$LimbLength(j)=(d_{i,j}+d_{j,l}-(d_{i,Parent(j)}+d_{l,Parent(j)}))/2\leq(d_{i,j}+d_{j,l}-d_{i,l})/2=(D_{i,j}+D_{j,l}-D_i)/2.$$

As a result of this discussion, $LimbLength(j)$ must be less than or equal to $(D_{i,j}+D_{j,k}-D_{i,k})/2$ for any choice of leaves i and k. Because we can always find leaves i and k belonging to different subtrees (why?), it follows that $LimbLength(j)$ is equal to the minimum value of $(D_{i,j}+D_{j,k}-D_{i,k})/2$ over all choices of i and k.

4. ADDITIVE PHYLOGENY

Trimming the tree

Since we now know how to find the length of any limb in $Tree(D)$, we can construct $Tree(D)$ recursively using the algorithm illustrated in the figure below and described on the next few steps.

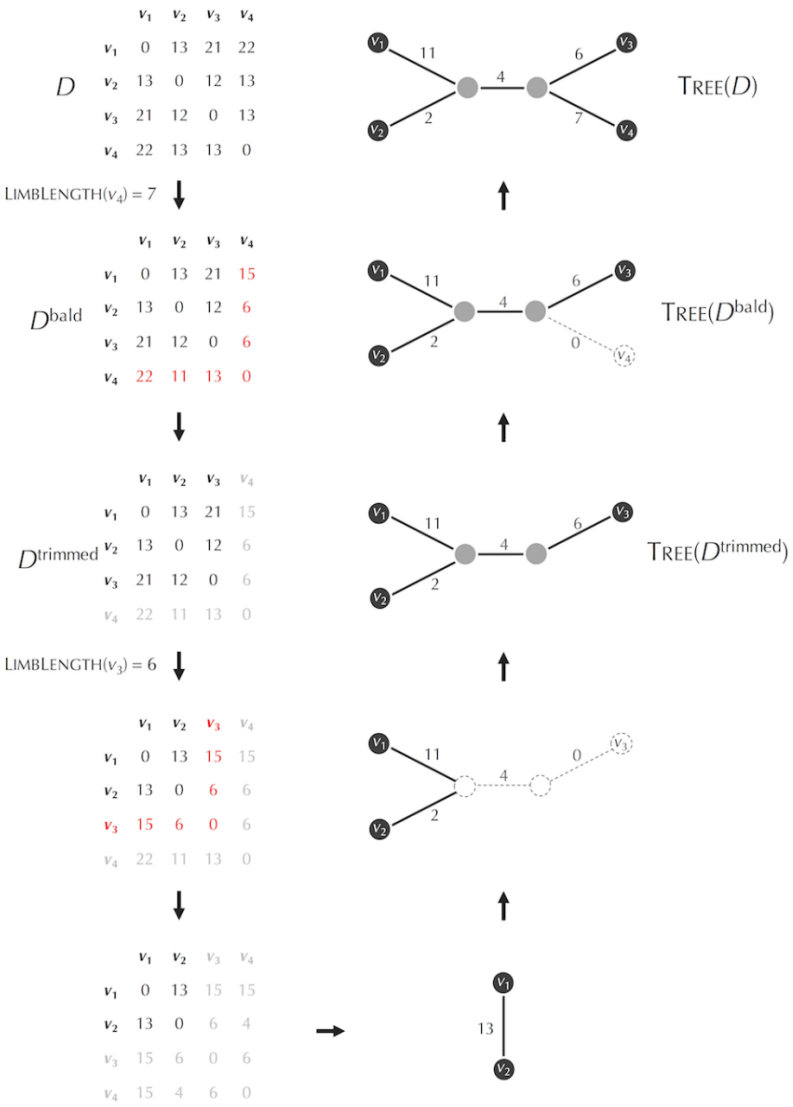


Figure: Converting an additive distance matrix into the simple tree fitting this matrix. On the left side, we first compute *LimbLength*(v_4) = 7, and then subtract 7 from the non-diagonal elements in the final row/column of D to obtain D^{bald} (updated values are shown in red). Removing this row and column yields a 3 x 3 distance matrix $D^{trimmed}$. We find that *LimbLength*(v_3) = 6 in $D^{trimmed}$ and subtract 6 from the non-diagonal elements in the third row and column. Graying out this row and column yields a 2 x 2 distance matrix. On the right side, we can fit this 2 x 2 distance matrix to a tree consisting of a single edge. By finding the attachment points of removed limbs (shown on the left), we reconstruct *Tree*($D^{trimmed}$), *Tree*(D^{bald}), and then *Tree*(D).

First, imagine that we already know *Tree*(D), and pick an arbitrary leaf j. We will trim the limb of j by reducing its length by *LimbLength*(j). Because we do not know *Tree*(D), we need to represent trimming the leaf j in terms of the distance matrix D. To do so, we first subtract *LimbLength*(j) from each off-diagonal element in row j and column j of D to obtain a matrix D^{bald} for which the limb of j has become a **bald limb**, or a limb of length 0 (see the figure

on the previous step). We will further assume that a bald limb has disappeared from the tree entirely. In terms of the distance matrix, ignoring a bald limb means removing row j and column j from D to produce a smaller $(n - 1) \times (n - 1)$ distance matrix $D^{trimmed}$.

We can now recursively find $Tree(D)$ in four steps:

- pick an arbitrary leaf j, compute $LimbLength(j)$, and construct the distance matrix $D^{trimmed}$;
- solve the Distance-Based Phylogeny Problem for $D^{trimmed}$;
- identify the point in $Tree(D^{trimmed})$ where leaf j should be attached in $Tree(D)$;
- add a limb of length $LimbLength(j)$ growing from this attachment point in $Tree(D^{trimmed})$ to form $Tree(D)$.

***Code challenge:** Write a pseudocode to recursively find Tree (D).

Question 4: When adding leaf j back to $Tree(D^{trimmed})$, how would you find its attachment point?

Attaching a limb

To find the attachment point of a leaf j in $Tree(D^{trimmed})$, consider $Tree(D^{trimmed})$, which is the same as $Tree(D)$ except that $LimbLength(j) = 0$. From the Limb Length Theorem, we know that there must be leaves i and k in $Tree(D^{trimmed})$ such that

$$(D^{bald}_{i,j}+D^{bald}_{j,k}-D^{bald}_{i,k})/2=0,$$

which implies that

$$D^{bald}_{i,k} = D^{bald}_{i,j} + D^{bald}_{j,k}$$

Thus, the attachment point for leaf j must be located at distance $D^{bald}_{i,j}$ from leaf i on the path connecting i and k in the trimmed tree. This attachment point may occur at an existing node, in which case we connect j to this node. On the other hand, the attachment point for j may occur along an edge, in which case we place a new node at the attachment point and connect j to it.

An algorithm for distance-based phylogeny reconstruction

The preceding discussion results in the following recursive algorithm, called **Additive Phylogeny**, for finding the simple tree fitting an $n \times n$ additive distance matrix D. It implements a program **Limb**(D, j) that computes $LimbLength(j)$ for a leaf j based on the distance matrix D.

Rather than selecting an arbitrary leaf *j* from *Tree*(*D*) for trimming, **Additive Phylogeny** selects leaf *n* (corresponding to the last row and column of *D*).

```
AdditivePhylogeny(D, n)
    if n = 2
        return the tree consisting of a single edge of length D1,2
    limbLength ← Limb(D, n)
    for j ← 1 to n - 1
        Dj,n ← Dj,n - limbLength
        Dn,j ← Dj,n
    (i, k) ← two leaves such that Di,k = Di,n + Dn,k
    x ← Di,n
    remove row n and column n from D
    T ← AdditivePhylogeny(D, n - 1)
    v ← the (potentially new) node in T at distance x from i on the
path between i and k
    add leaf n back to T by creating a limb (v, n) of length
limbLength
    return T
```

Question 6: Consider these questions about **AdditivePhylogeny**.

- Although it may seem that **AdditivePhylogeny** would construct a tree for any matrix, this is not the case. What goes wrong if you apply **AdditivePhylogeny** to the non-additive distance matrix in the figure below?

- Modify **AdditivePhylogeny** to develop an algorithm that checks whether a given distance matrix is additive. Then, apply this test to the distance matrix for coronavirus Spike proteins shown on the next step. Is this matrix additive?

	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

Constructing an evolutionary tree of coronaviruses

By the end of 2003, bioinformaticians had sequenced many coronaviruses from a variety of birds and mammals, from which we obtain the distance matrix below based on a multiple alignment of Spike proteins.

Table: The distance matrix based on pairwise alignment of Spike proteins from coronaviruses extracted from various animals. The distance between each pair of sequences was computed as the total number of mismatches and indels in their optimal alignment.

	Cow	Pig	Horse	Mouse	Dog	Cat	Turkey	Civet	Human
Cow	0	295	300	524	1077	1080	978	941	940
Pig	295	0	314	487	1071	1088	1010	963	966
Horse	300	314	0	472	1085	1088	1025	965	956
Mouse	524	487	472	0	1101	1099	1021	962	965
Dog	1076	1070	1085	1101	0	818	1053	1057	1054
Cat	1082	1088	1088	1098	818	0	1070	1085	1080
Turkey	976	1011	1025	1021	1053	1070	0	963	961
Civet	941	963	965	962	1057	1085	963	0	16
Human	940	966	956	965	1054	1080	961	16	0

Although you now understand the perils of concluding that the minimum element of the distance matrix corresponds to a pair of neighbors, common sense tells us with a glance at the coronavirus distance matrix that the civet must be the animal reservoir of SARS. This information led researchers to hypothesize that inadequate preparation of meat from palm civets (see the photo below) in the Guangdong region of China may have caused the SARS outbreak.

Figure: The palm civet.

Yet before rushing to this conclusion, we note that some studies have suggested that humans first received SARS from bats, which later gave the virus to palm civets, which then transmitted the disease back to humans. The civet was identified as the animal reservoir of SARS in 2003 in part because SARS viruses from other potential suspects, including bats, had not yet been sequenced.

Since the distance matrix for SARS-like coronaviruses is non-additive, we will cheat a bit and slightly modify it to make it additive so that you can apply **Additive Phylogeny** to it (see the figure below).

Exercise 7: Construct the simple tree fitting this distance matrix.

Table: A modification of the Spike protein distance matrix to make it additive.

	Cow	Pig	Horse	Mouse	Dog	Cat	Turkey	Civet	Human
Cow	0	295	306	497	1081	1091	1003	956	954
Pig	295	0	309	500	1084	1094	1006	959	957
Horse	306	309	0	489	1073	1083	995	948	946
Mouse	497	500	489	0	1092	1102	1014	967	965
Dog	1081	1084	1073	1092	0	818	1056	1053	1051
Cat	1091	1094	1083	1102	818	0	1066	1063	1061
Turkey	1003	1006	995	1014	1056	1066	0	975	973
Civet	956	959	948	967	1053	1063	975	0	16
Human	954	957	946	965	1051	1061	973	16	0

5. USING LEAST SQUARES TO CONSTRUCT APPROXIMATE PHYLOLGENIES

If an $n \times n$ distance matrix D is non-additive, then we will instead look for a weighted tree T whose distances between leaves approximate the entries in D. To this end, we would like for T to minimize the **sum of squared errors** *Discrepancy*(T, D), which is given by the formula

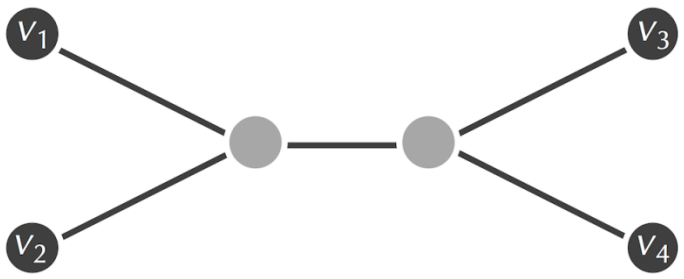
$$Discrepancy(T,D) = \sum_{1 \le i < j \le n} (d_{i,j}(T) - D_{i,j})^2$$

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

Input: An $n \times n$ distance matrix D.

Output: A weighted tree T minimizing *Discrepancy*(T, D) over all weighted trees with n leaves.

Exercise 8: Let T be the tree shown below.



Given the non-additive 4 × 4 distance matrix *D* below, find the lengths of edges in this tree that minimize *Discrepancy*(*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

It turns out that for a specific tree *T*, it is easy to find edge lengths in *T* minimizing *Discrepancy*(*T*, *D*). Yet our ability to minimize the sum of squared errors for a specific tree does not imply that we can efficiently solve the Least Squares Distance-Based Phylogeny Problem, since the number of different trees grows very quickly as the number of leaves in the tree increases. In fact, the Least Squares Distance-Based Phylogeny winds up being *NP*-Complete, and so we must abandon the hope of designing a fast algorithm to find a tree that best fits a non-additive matrix. In the next two sections, we will explore heuristics for constructing trees from non-additive matrices that solve this problem approximately.

6. ULTRAMETRIC EVOLUTIONARY TREES

Biologists often assume that every internal node in an evolutionary tree corresponds to a species that underwent a **speciation event**, splitting one ancestral species into two descendants. Note that every internal node in the tree in the figure below (corresponding to a speciation event) has degree 3. We therefore define an **unrooted binary tree** as a tree where every node has degree equal to either 1 or 3.

Exercise 9: Prove that every unrooted binary tree with *n* leaves has *n* - 2 internal nodes (and thus 2*n* - 3 edges).

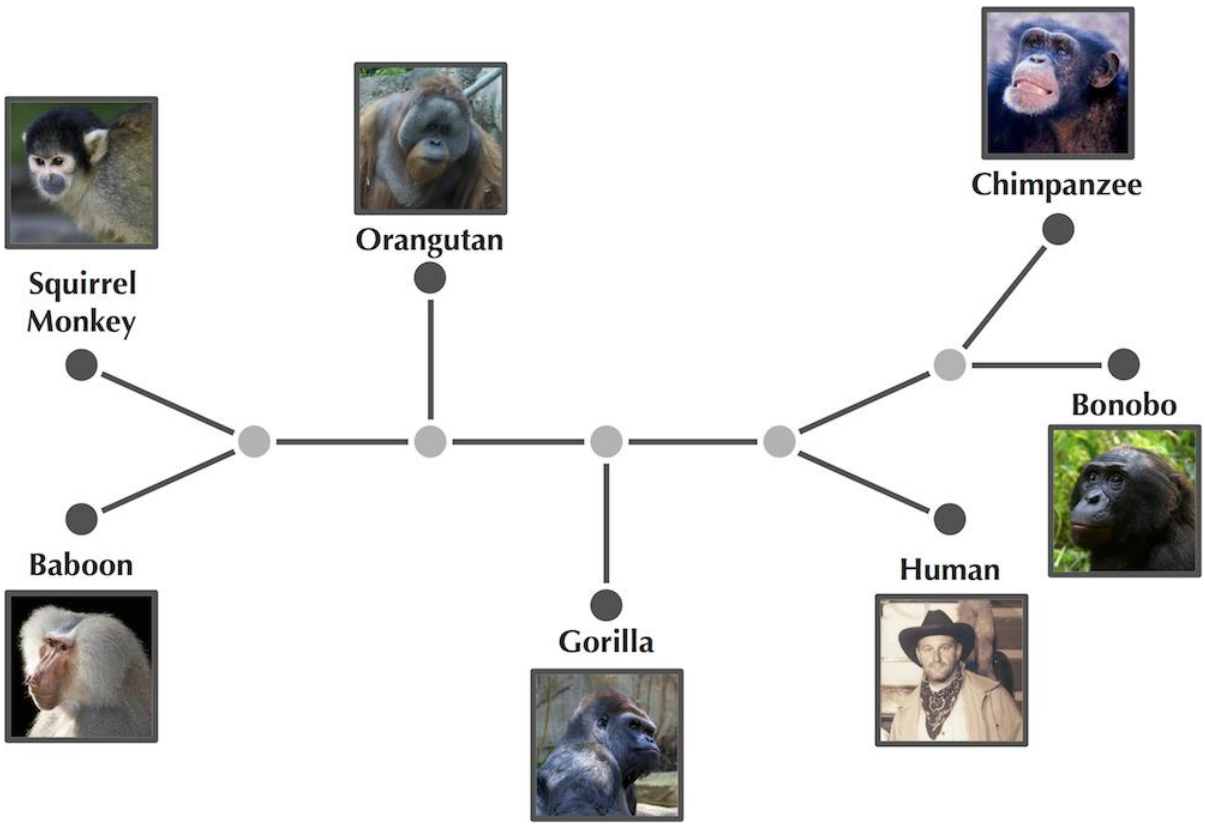


Figure: An unrooted binary tree modeling the primate phylogeny.

A **rooted binary tree** is an unrooted binary tree that has a root (of degree 2) placed on one of its edges; in other words, we replace an edge (*v*, *w*) with a root and draw edges connecting the root to each of *v* and *w*.

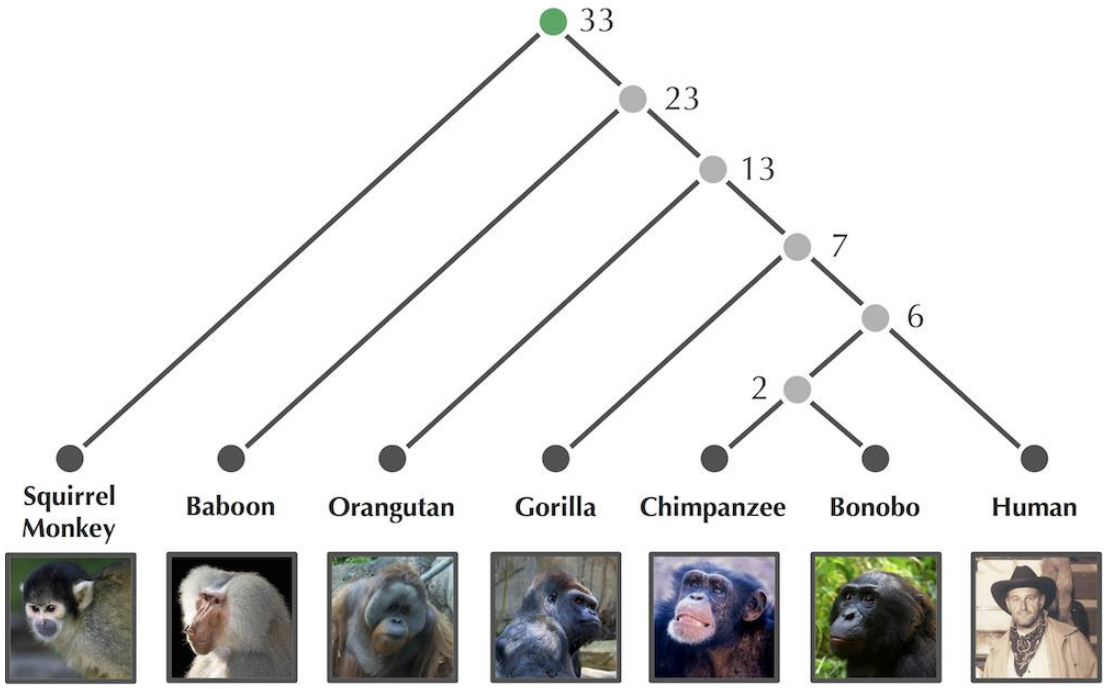


Figure: Placing a root on the squirrel monkey's limb results in a rooted binary tree. The number at each node corresponds to the number of million years ago that the divergence at this node occurred.

If we had a **molecular clock** measuring evolutionary time, then we could assign an **age** to every node *v* in a rooted binary tree (denoted *Age*(*v*)), where all of the leaves of the tree have age 0 because they correspond to present-day species. We could then define the weight of an edge (*v*, *w*) in the tree as the difference *Age*(*v*) - *Age*(*w*). Consequently, the length of a path between the root and any node would be equal to the difference between their ages. Such a tree, in which the distance from the root to any leaf is the same, is called **ultrametric** (see figure below).

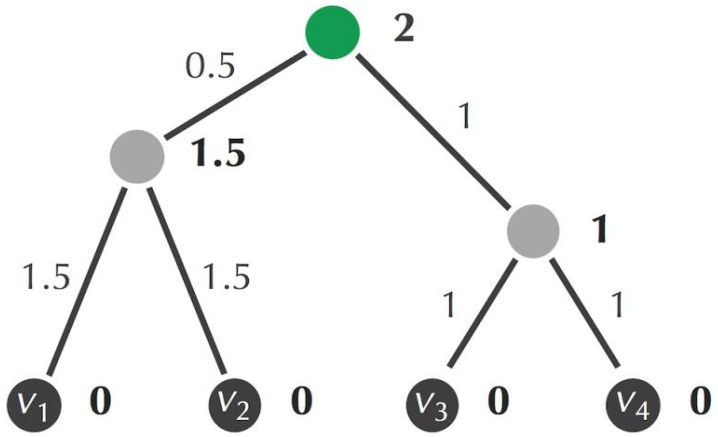


Figure: An ultrametric tree. Each node has been assigned an age (in boldface), where each leaf has age equal to zero. The length of each edge is equal to the difference between the ages of the nodes that the edge connects, and the distance from the root to any leaf is equal to 2, the age of the root.

Our aim is to derive an ultrametric tree that explains a given distance matrix (even if it does so only approximately). **UPGMA** (which stands for **U**nweighted **P**air **G**roup **M**ethod with **A**rithmetic Mean) is a simple clustering heuristic that introduces a hypothetical molecular clock for constructing an ultrametric evolutionary tree.

Given an $n \times n$ matrix D, **UPGMA** (which is illustrated in the figure on the next step) first forms n trivial clusters, each containing a single leaf. The algorithm then finds a pair of "closest" clusters. To clarify the notion of closest clusters, **UPGMA** defines the distance between clusters C_1 and C_2 as the average pairwise distance between elements of C_1 and C_2:

$$D_{C1,C2}=\sum_{i\in C1}\sum_{j\in C2} D_{i,j} / \mid C_1|\cdot|C_2|.$$

In this equation, the notation $|C|$ denotes the number of leaves in cluster C.

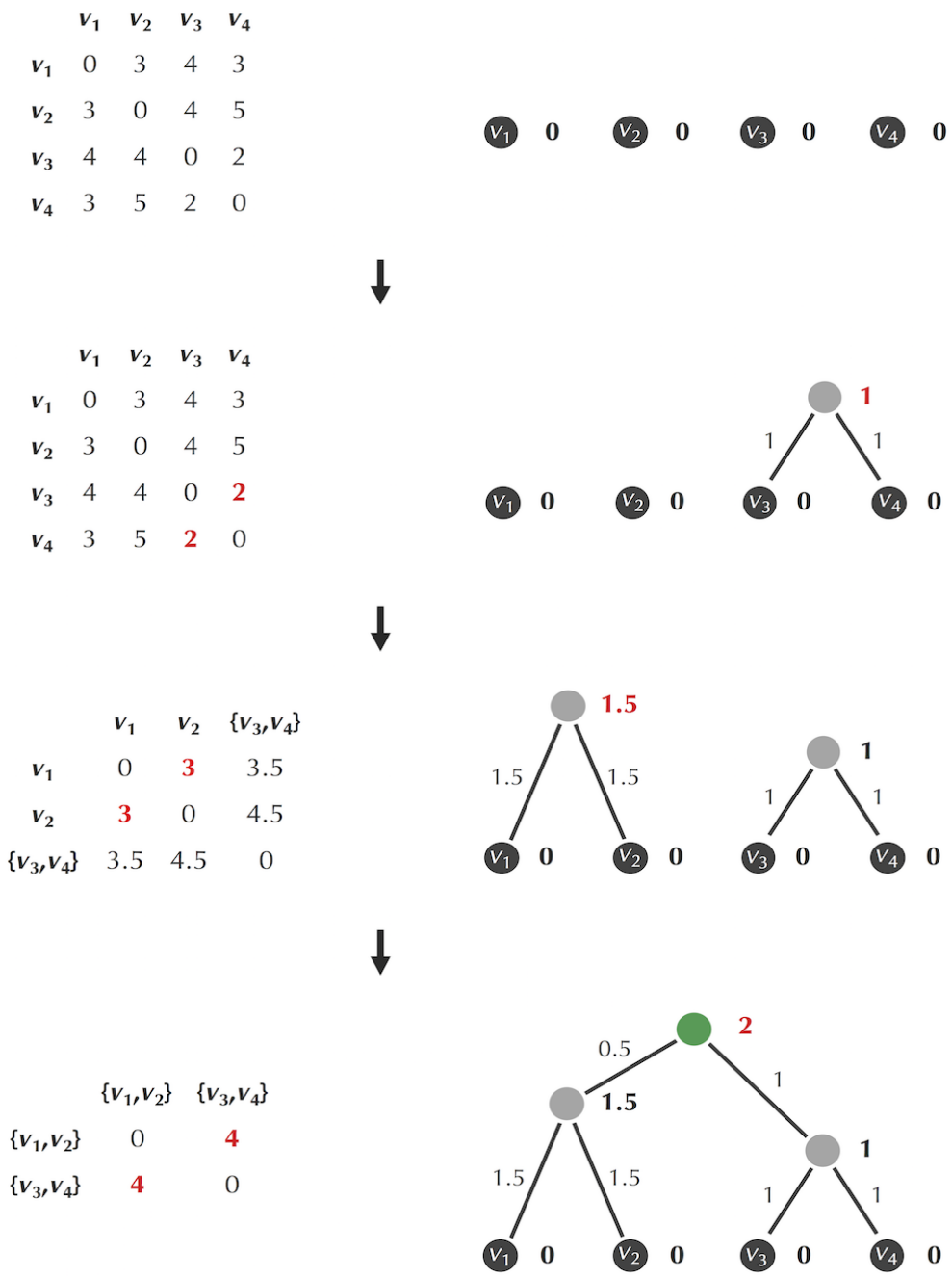


Figure: Tree reconstruction with **UPGMA** for the non-additive distance matrix we have encountered previously. **UPGMA** begins with forming one cluster for each leaf. In each step, it identifies the two closest clusters C_1 and C_2, merge them into a new node C, and connect C to C_1 and C_2 by directed edges. The age of C is set equal to $D_{C1,C2}$ / 2. We then iterate this process until only a single cluster remains, which must be the root.

Once **UPGMA** has identified a pair of closest clusters C_1 and C_2, it merges them into a cluster C with $|C_1| + |C_2|$ elements and then creates a node for C, which it connects to each of C_1 and C_2 by a directed edge. The age of C is set to be $D_{C1,C2}$ /2. **UPGMA** then iterates this process of merging the two closest clusters until only a single cluster remains, which corresponds to the root.

Exercise 10: Prove that after merging clusters C_i and C_j into a cluster C_{new}, the distance between C_{new} and another cluster C_m is equal to $(D_{Ci,Cm} \cdot |C_i| + D_{Cj,Cm} \cdot |C_j|) / (|C_i| + |C_j|)$.

The pseudocode for **UPGMA** is shown below.

```
UPGMA(D, n)
      Clusters ← n single-element clusters labeled 1, ... , n
     construct a graph T with n isolated nodes labeled by single
elements 1, ... , n
    for every node v in T
        Age(v) ← 0
    while there is more than one cluster
        find the two closest clusters C_i and C_j
        merge C_i and C_j into a new cluster C_new with |C_i| + |C_j| elements
        add a new node labeled by cluster C_new to T
        connect node C_new to C_i and C_j by directed edges
        Age(C_new) ← D_{Ci, Cj} / 2
        remove the rows and columns of D corresponding to C_i and C_j
        remove C_i and C_j from Clusters
        add a row/column to D for C_new by computing D(C_new, C) for each C
in Clusters
        add C_new to Clusters
    root ← the node in T corresponding to the remaining cluster
    for each edge (v, w) in T
        length of (v, w) ← Age(v) - Age(w)
    return T
```

For concreteness, this example might be helpful in explaining the pseudocode:

https://en.wikipedia.org/wiki/UPGMA#Working_example

Exercise 11: Apply **UPGMA** to the (non-additive) coronavirus distance matrix reproduced below.

Cow	Pig	Horse	Mouse	Dog	Cat	Turkey	Civet	Human	
Cow	0	295	300	524	1077	1080	978	941	940
Pig	295	0	314	487	1071	1088	1010	963	966
Horse	300	314	0	472	1085	1088	1025	965	956
Mouse	524	487	472	0	1101	1099	1021	962	965
Dog	1076	1070	1085	1101	0	818	1053	1057	1054
Cat	1082	1088	1088	1098	818	0	1070	1085	1080
Turkey	976	1011	1025	1021	1053	1070	0	963	961
Civet	941	963	965	962	1057	1085	963	0	16

Human 940 966 956 965 1054 1080 961 16 0

UPGMA offers a step forward from **AdditivePhylogeny**, since it can analyze non-additive distance matrices. The figure below shows the result of applying **UPGMA** to the coronavirus distance matrix. However, the first step that **UPGMA** takes is to merge the two leaves *i* and *j* with minimum distance $D_{i,j}$ into a single cluster. We have already seen that the smallest element in the distance matrix does not necessarily correspond to a pair of neighboring leaves! This is a concern, since if **UPGMA** generates incorrect trees from additive matrices, then it is not an ideal heuristic for evolutionary tree construction from non-additive matrices. Can we find an algorithm that always identifies neighboring leaves in an additive distance matrix but also performs well on a non-additive distance matrix?

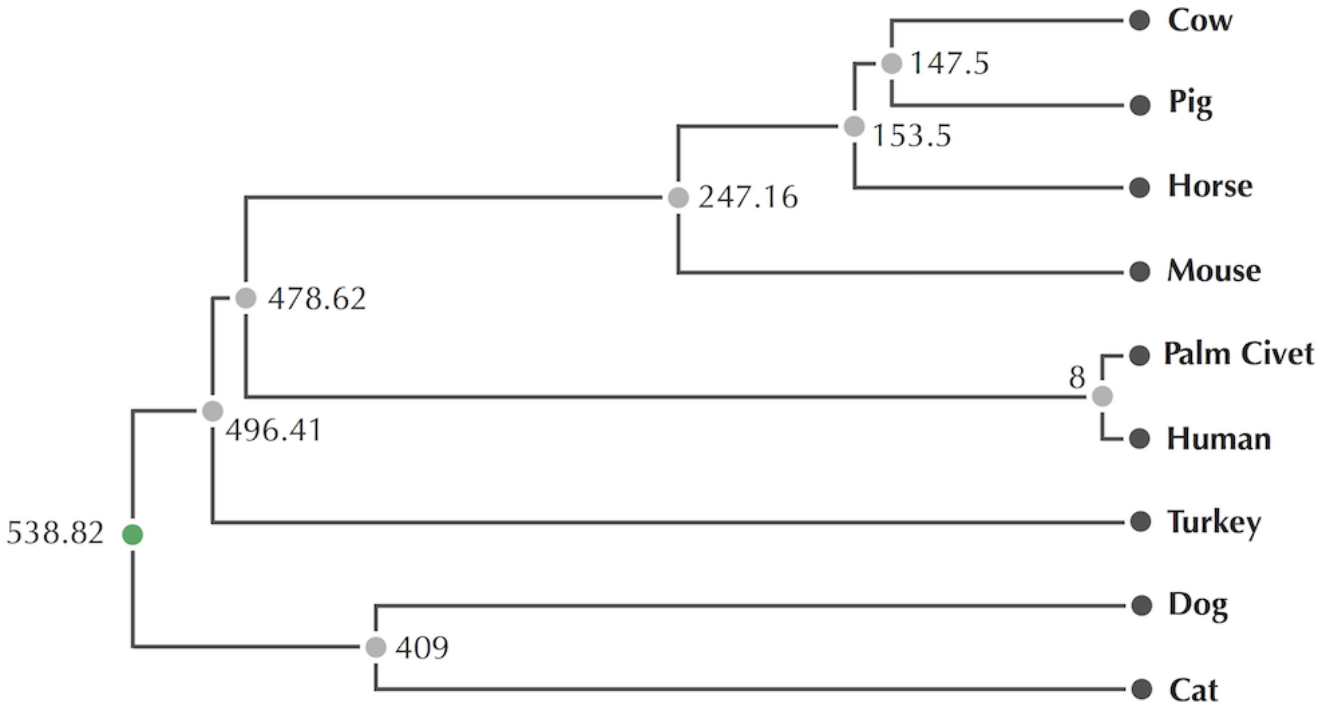


Figure: The ultrametric tree of coronaviruses created by **UPGMA** using the coronavirus distance matrix. The root is shown in green.

Transforming a distance matrix into a neighbor-joining matrix

In 1987, Naruya Saitou and Masatoshi Nei developed the **neighbor-joining algorithm** for evolutionary tree reconstruction. Given an additive distance matrix, this algorithm, which we call **NeighborJoining**, finds a pair of neighboring leaves and substitutes them by a single leaf, thus reducing the size of the tree. **NeighborJoining** can thus recursively construct a tree fitting the

additive matrix. This algorithm also provides a heuristic for non-additive distance matrices that performs well in practice.

The central idea of **NeighborJoining** is that although finding a minimum element in a distance matrix D is not guaranteed to yield a pair of neighbors in $Tree(D)$, we can convert D into a different matrix whose minimum element does yield a pair of neighbors. First, given an $n \times n$ distance matrix D, we define $\text{TotalDistance}_D(i)$ as the sum $\sum_{1 \le k \le n} D_{i,k}$ of distances from leaf i to all other leaves. The **neighbor-joining matrix** D^* (see below) is defined such that for any i and j, $D^*_{i,i} = 0$ and

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

D

	i	*j*	*k*	*l*	$TotalDistance_D$
i	0	13	21	22	56
j	13	0	12	13	38
k	21	12	0	13	46
l	22	13	13	0	48

D*

	i	*j*	*k*	*l*
i	0	-68	-60	-60
j	-68	0	-60	-60
k	-60	-60	0	-68
l	-60	-60	-68	0

NeighborJoining (illustrated on the next step) is a widely used method for evolutionary tree reconstruction; the paper that introduced it is one of the most cited in all of science, with over 30,000 citations. Yet this algorithm is non-intuitive: the above formula for computing the matrix D^* probably looks like witchcraft to you. In fact, despite having flawless intuition, Saitou and Nei *never proved* that their algorithm correctly solves the Distance-Based Phylogeny Problem for additive matrices!

Neighbor-Joining Theorem: Given an additive matrix D, the smallest element $D^*_{i,j}$ of its neighbor-joining matrix D^* reveals a pair of neighboring leaves i and j in $Tree(D)$.
If $n = 2$, then $NeighborJoining(D, n)$ returns the tree consisting of a single edge of length $D_{1,2}$. If $n > 2$, then it selects the minimum element in the neighbor-joining matrix, replaces the neighboring leaves i and j with a new leaf m, and then computes the distance from m to any other leaf k according to the formula

$$D_{k,m} = (1/2)(D_{k,i} + D_{k,j} - D_{i,j}).$$

which is motivated by **Additive Phylogeny**. This equation allows us to replace an $n \times n$ matrix D with an $(n - 1) \times (n - 1)$ matrix D' in which i and j have been replaced by m. By recursively applying *NeighborJoining* to D', we obtain an evolutionary tree with $n - 1$ leaves. We then add two limbs starting at node m, one ending in leaf i and the other ending in leaf j. We set

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

and assign

$$LimbLength(i) = (1/2) (D_{i,j} + \Delta_{i,j})$$
$$LimbLength(j) = (1/2) (D_{i,j} - \Delta_{i,j})$$

Exercise 12: Prove that if D is additive, then for any i and j between 1 and n, both $(1/2) (D_{i,j} + \Delta_{i,j})$ and $(1/2) (D_{i,j} - \Delta_{i,j})$ are non-negative.

The pseudocode below summarizes the neighbor-joining algorithm.

```
NeighborJoining(D,n)
    if n = 2
        T ← tree consisting of a single edge of length D1,2
        return T
    D* ← neighbor-joining matrix constructed from the distance matrix
D
    find elements i and j such that D*i,j is a minimum non-diagonal
element of D*
    Δ ← (TotalDistanceD(i) - TotalDistanceD(j)) /(n - 2)
    limbLengthi ← (1/2)(Di,j + Δ)
    limbLengthj ← (1/2)(Di,j - Δ)
    add a new row/column m to D so that Dk,m = Dm,k = (1/2)(Dk,i + Dk,j - Di,j)
for any k
    remove rows  i and j from D
    remove columns i and j from D
    T ← NeighborJoining(D, n - 1)
    add two new limbs (connecting node m with leaves i and j) to the
tree T
    assign length limbLengthi to Limb(i)
    assign length limbLengthj to Limb(j)
    return T
```

Exercise 13: Before implementing the neighbor-joining algorithm, try applying it to the additive and non-additive distance matrices shown below.

	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

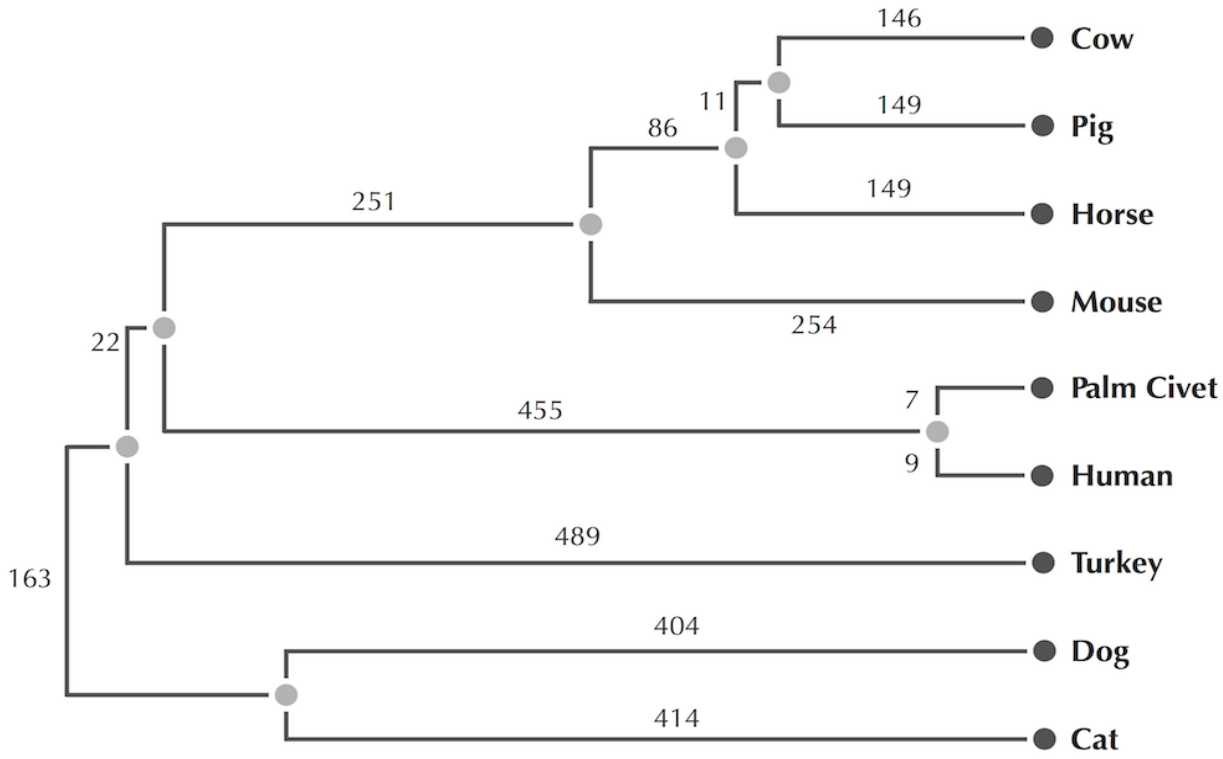
	v_1	v_2	v_3	v_4
v_1	0	3	4	3
v_2	3	0	4	5
v_3	4	4	0	2
v_4	3	5	2	0

Exercise 14: Apply **Neighbor-Joining** to the non-additive coronavirus distance matrix, reproduced below.

Cow	Pig	Horse	Mouse	Dog	Cat	Turkey	Civet	Human	
Cow	0	295	300	524	1077	1080	978	941	940
Pig	295	0	314	487	1071	1088	1010	963	966
Horse	300	314	0	472	1085	1088	1025	965	956
Mouse	524	487	472	0	1101	1099	1021	962	965
Dog	1076	1070	1085	1101	0	818	1053	1057	1054
Cat	1082	1088	1088	1098	818	0	1070	1085	1080
Turkey	976	1011	1025	1021	1053	1070	0	963	961
Civet	941	963	965	962	1057	1085	963	0	16
Human	940	966	956	965	1054	1080	961	16	0

Analyzing coronaviruses with the neighbor-joining algorithm

The figure below shows the neighbor-joining tree of coronaviruses isolated from different animals based on the nonadditive distance matrix from the previous step. (Edge-weights are rounded to the nearest integer.)



We can also apply **Neighbor-Joining** to the distance matrix of SARS-CoV variants isolated from various human carriers, in addition to a coronavirus from palm civet (see the figure below). The SARS-CoV strain labeled "Hanoi" in this figure was taken from **Carlo Urbani**, an Italian physician who worked for the World Health Organization. In February 2003, Urbani was called into a Hanoi hospital to examine a patient who had fallen ill with what local doctors believed to be a bad case of influenza; these doctors were afraid that it might be bird flu. In fact, the patient was the American man who had stayed across the hall from Liu Jianlun at the Metropole Hotel just one week earlier. Fortunately, Urbani was quick to realize that the disease was not influenza, and he became the first physician to raise the alarm to public health officials. Yet rather than leaving Hanoi, he demanded that he remain there in order to oversee quarantine procedures. In an argument with his wife, who scolded him for risking his life to treat sick patients. Urbani replied, "What am I here for? Answering e-mails, going to cocktail parties and pushing paper?" Urbani would ultimately lose his life to SARS a month later. But his sacrifice also helped begin the massive worldwide action against this disease that may well have saved millions of lives.

	Guangzhou Dec. 16, 2002	Zhonghan Dec.16, 2002	Guangzhou Jan. 24, 2003	Guangzhou Jan. 31, 2003	Guangzhou Feb. 18, 2003	Hong Kong Feb. 18, 2003	Hanoi Feb. 26, 2003	Toronto Feb. 17, 2003	Hong Kong March 15, 2003	Palm civet
Guangzhou	0	4	12	8	9	9	12	12	11	3
Zhonghan	4	0	10	6	7	7	10	10	9	3
Guangzhou	12	10	0	4	5	3	2	2	1	11
Guangzhou	8	6	4	0	3	1	4	4	3	7
Guangzhou	9	7	5	3	0	2	5	5	4	8
Hong Kong	9	7	3	1	2	0	3	3	2	8
Hanoi	12	10	2	4	5	3	0	2	1	11
Toronto	12	10	2	4	5	3	2	0	1	11
Hong Kong	11	9	1	3	4	2	1	1	0	10
Palm civet	3	3	11	7	8	8	11	11	10	0

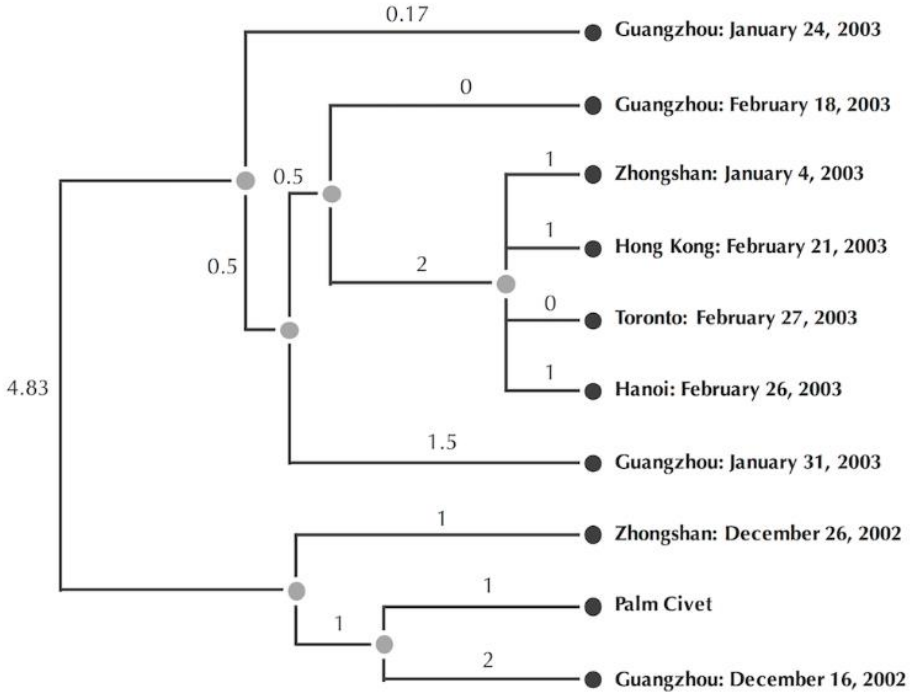


Figure: (Top) The distance matrix based on pairwise alignment of Spike proteins from SARS-CoV strains extracted from various patients as well as a coronavirus taken from a palm civet. The distance between each pair of sequences was computed as the total number of mismatches and indels in their optimal pairwise alignment. (Bottom) The evolutionary tree of these viruses constructed by the neighbor-joining algorithm.

Limitations of distance-based approaches to evolutionary tree reconstruction

Although distance-based tree reconstruction has successfully resolved questions about the origin and spread of SARS, many evolutionary controversies cannot be resolved by using a distance matrix. For example, when we convert each pair of rows of a multiple alignment into a distance value, we lose information contained in the alignment. As a result, distance-based methods do not allow us to reconstruct ancestral sequences of Spike proteins (corresponding to internal nodes of the Spike protein phylogeny), which could lead us to believe that such molecular paleontology is impossible. Therefore, a superior approach to evolutionary tree reconstruction would be to somehow use the alignment directly, without first converting it to a distance matrix.