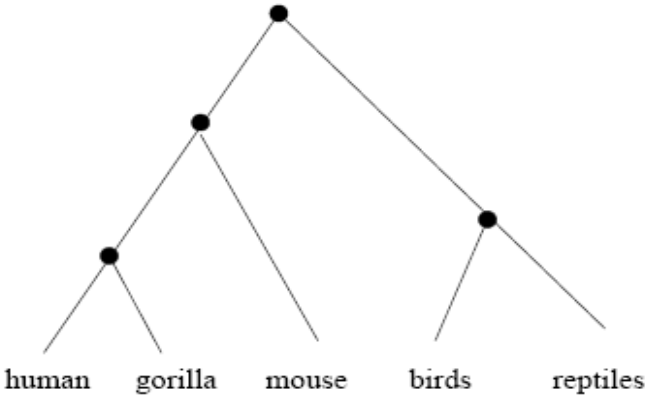
Phylogenic Trees

In any evolutionary process, *speciation* events cause a new species to split off from an existing one, thus creating the diversity of life forms we know today.



A key issue in evolutionary biology is to reconstruct the history of these speciation events. Given the properties of the leaf nodes, reconstruct what the tree is.

Much of the current interest in phylogenic trees follows from the increasing availability of DNA sequence data.

Biological applications include evolution studies (e.g. the *out of Africa debate*) and medical research (tracing HIV infection).

However, phylogenic trees play an important role in analyzing the history of languages, religions, chain letters, and medieval manuscripts, as well as biology.

What data is available?

Although the available for analysis varies by application, it can usual be partitioned into *distance* and *feature/character* data.

Distance data measures (directly or indirectly) d_{ij}, the length of time since species i and j diverged. Such time can be estimated from the distance between DNA sequences, assuming a 'molecular clock' governing the frequency of mutations.

Feature/character data measures taxonomical properties such as 'warm blooded', 'has wings', or 'walks upright'. If such features are hard to develop twice, they describe branch points in the phylogenic tree.

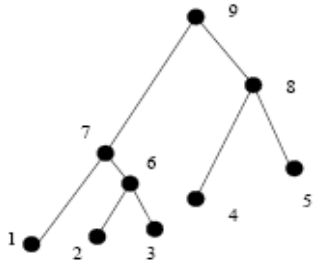
Different types of reconstruction algorithms are necessary for such distinct types of data.

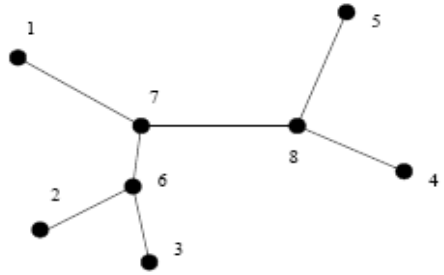
Counting Trees

Observe that there are many tree *topologies* possible for any set of n leaves.

Every *binary* tree on n leaves has $n-1$ *internal* nodes, and thus $2n-1$ vertices and $2n-2$ edges.

Every *unrooted* binary tree n leaves has $2n-2$ vertices and $2n-3$ edges, since the in-degree 0 root can be contacted to a single edge.





Since the root can be positioned at any edge, there are $2n-3$ more rooted trees than unrooted trees on n leaves.

Further, any rooted tree on n leaves corresponds to an unrooted tree on $n+1$ leaves, since we can take the highest numbered leaf to be the root.

Tree counting recurrence

Thus

$$Rooted[n] = (2n-3)Unrooted[n]$$

$$Rooted[n] = Unrooted[n+1]$$

yielding that

$$Rooted[n] = (2n-3)(2n-5)\ldots 1$$

For $n = 10$, there are about 2,000,000 unrooted trees, and for $n = 20$ about 2.2×10^{20}, so the number of possible topologies grows very fast.

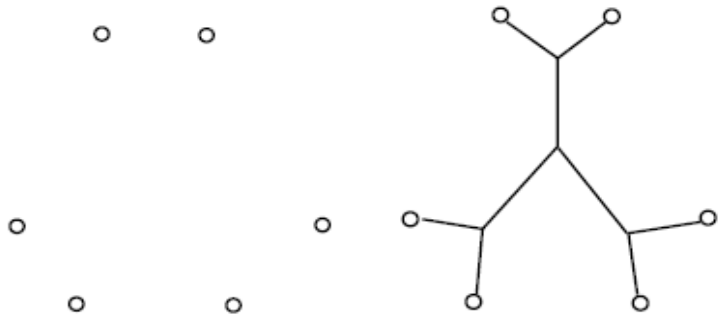
This makes it hopeless to exhaustively search for the best possible tree beyond 10 or so species.

Why is it difficult?

The business of reconstructing trees is very messy for several reasons:

- Nobody knows the correct underlying trees, but many people have strong and contradictory opinions.
- The data is inherently noisy, error-prone, and hard to interpret.
- Most precise definitions of optimal trees lead to NP-complete problems soon as errors creep into the data.
- Many different tree reconstruction algorithms and heuristics have been proposed, each of which yields different trees on the same data.

Phylogenic tree problems have the same flavor as *Steiner* tree problems in graphs, where we must deduce the positions of intermediate nodes to find the best possible fit.



Algorithms for distance data

Distance data tree construction algorithms bare a strong relationship to *clustering algorithms*, particularly agglomerative algorithms which explicitly construct a tree as they merge clusters together.

Representative algorithms include *nearest neighbor* joining, and repeatedly merging the nearest cluster centroids (the *unweighted pair group method using arithmetic averages* (UPGMA)).

Such algorithms can yield reasonable and informative trees, but there seems no good reason to believe that they yield *the* correct tree.

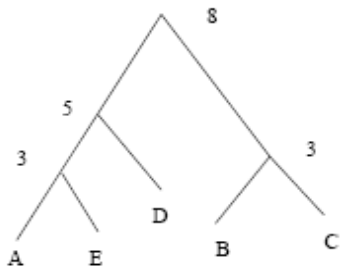
Defining mathematical properties which real trees obey enable us to define optimization criteria which make it plausible to define the *best* tree.

Ultrametric trees

An *ultrametric* tree is a rooted tree where

- Each internal node is labeled by a number.
- Each internal node has at least two children
- Along any root to leaf path the labels strictly decrease.

Note that if the labels mean "time units ago", this is true of any evolutionary tree.



	A	B	C	D	E
A		8	8	5	3
B			3	8	8
C				8	8
D					5
E					

Each pairwise distance $d(i, j)$ represents the label of the least common ancestor of i and j.

Reconstructing ultrametric trees

There is an efficient $O(n^2)$ algorithm to reconstruct an ultrametric tree, if one exists.

Observe that the labels on the path from a leaf a to the root follow from sorting the distances in row a, since a branching point corresponds to each unique distance.

Shared distances on the path partition the other leaves into groups in the other subtrees.

Thus each of the resulting partitions is fixed, and can be refined by considering other rows.

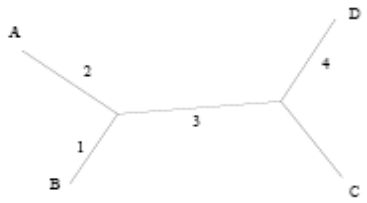
Unless we get a contradiction, we get an ultrametric tree. Further, this tree must be unique.

The problem becomes hard when you seek the "most" ultrametric tree in noisy data. A little noise can throw the topology off considerably.

Additive trees

A weaker but well defined condition assumes that we have the *distance* between all pairs of leaves, and seek an unrooted, edge-weighted tree such that the sum of the distances along each path adds up to the defined distance.

	A	B	C	D
A		3	7	9
B			6	8
C				6
D				



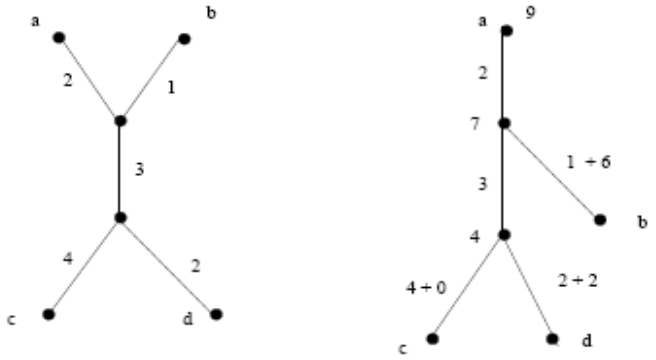
An algorithm for additive tree reconstruction follows from a reduction to ultrametric trees.

Additive trees

First, lift one of the nodes v with a maximum distance entry M_v to the root.

Second, add weight to each leaf edge i so that the total distance from root to each leaf is M_v.

Finally, assign each node the largest weight below it to give us matrix D':

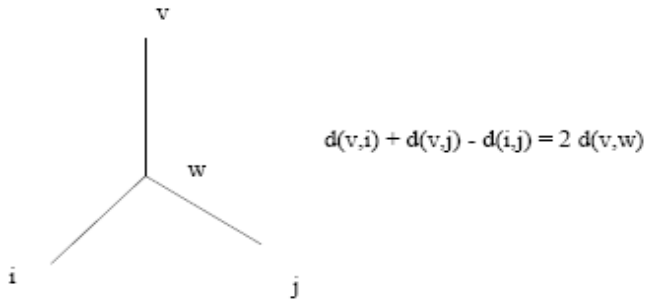


Additive trees

These node labels of D' define an ultrametric tree, since they decrease along each root-to-leaf path.

Thus if we could construct this matrix D' without the knowledge of the tree, we could solve the additive tree problem as an ultrametric tree problem by reversing this construction.

Since the additive input matrix D determines v and M_v, we need to compute the distance in the unknown tree from v to the *least common ancestor* $LCA(i, j)$.



Thus if D is an additive matrix, then D' is an ultrametric matrix, where

$$D' = M_v(D) + (D(i, j) - D(v, i) - D(v, j))/2$$

The technical assumption that all species in an ultrametric tree are leaves can be restored by hanging v directly off the root.

Additive trees

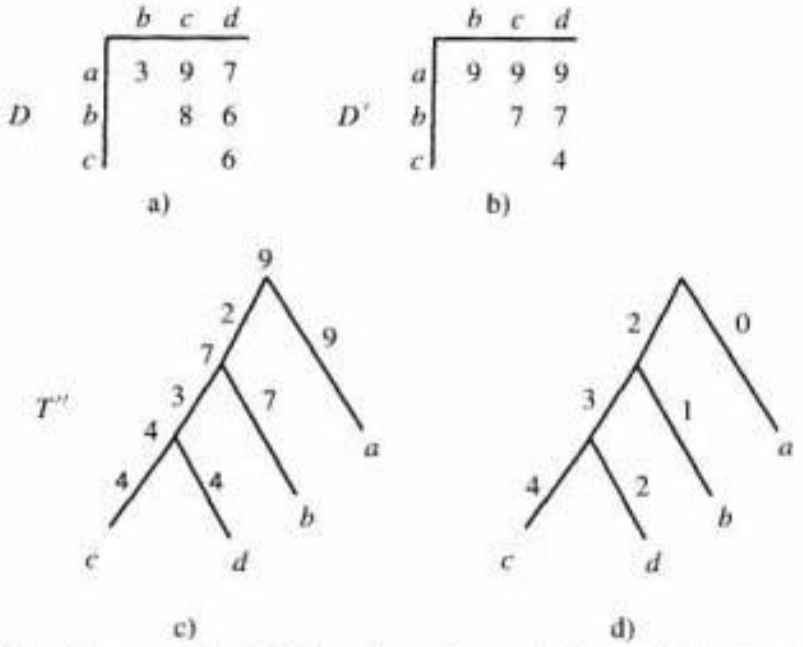


Figure 17.12: a. The distance matrix D obtained from the tree in Figure 17.10a. The largest entry has value 9 and is in row a. b. The derived ultrametric matrix D'. c. The ultrametric tree T'' along with the derived edge weights. d. The resulting tree after $m_a - D(a, i)$ is subtracted from leaf edges. The original tree is recovered after contracting the zero-weight edge to leaf a.

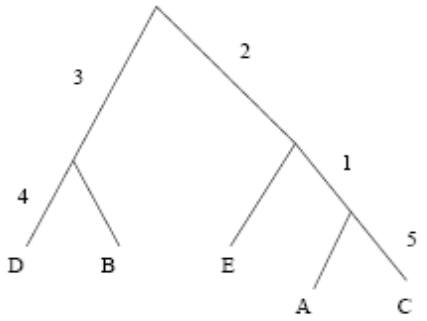
Perfect Phylogenies

Suppose we have n species, and m features each of which only evolved once (*parsimony*).

Each feature can be represented by an n-element $\{0, 1\}$-vector where $f_i(j) = 1$ iff species j has feature i.

The *perfect phylogeny problem* asks for a phylogenic tree given an $n \times m$ binary feature matrix, if one exists.

	1	2	3	4	5
A	1	1	0	0	0
B	0	0	1	0	0
C	1	1	0	0	1
D	0	0	1	1	0
E	0	1	0	0	0



Note that this is an edge-labeled tree, and certain features do not necessarily contribute to a split (e.g. 3), particularly when $m > n$.

Reconstructing Perfect Phylogenies

Suppose we know that all features evolve from 0 to 1, i.e. the characters are ordered.

Claim: Matrix M has a perfect phylogenic tree iff for every pair of columns i, j the set of 1s are either (1) disjoint, or (2) i contains j.

If column i *contains* all the 1s of column j, then feature j evolved *before* feature i.

If two columns are disjoint, then the features evolved *independently*.

If species x has feature i but not j, and y has a feature j but not i, then *no perfect phylogeny can exist*.

This immediately gives an $O(m^2n)$ algorithm to test the matrix for this property, which can be improved to $O(nm)$ using radix sorting.

Perfect phylogeny via ultrametric trees

Define an $n \times n$ matrix where $D[i, j]$ equals the number of characters which species i shares with j for a given feature matrix M.

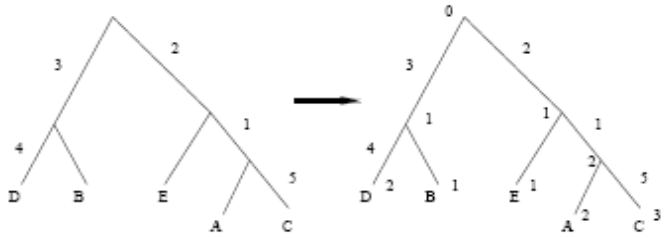
	1	2	3	4	5
A	1	1	0	0	0
B	0	0	1	0	0
C	1	1	0	0	1
D	0	0	1	1	0
E	0	1	0	0	0

→

	A	B	C	D	E
A	2	0	2	0	1
B		1	0	1	0
C			3	0	1
D				2	0
E					1

Perfect phylogeny via ultrametric trees

Given a perfect phylogeny for M, we can label each node of the tree with the number of labels encountered from the root:



These numbers are increasing along each root-to-leaf path, so negating them given an ultrametric matrix.

Thus if M has a perfect phylogeny, then D must be ultrametric.

This gives a reconstruction algorithm since ultrametric trees are unique.

Generalizing perfect phylogeny

The perfect phylogeny problem can be made more general by allowing non-binary features, e.g. the locomotion feature might be 'fixed', 'crawling', or 'walking'.

In general, each feature is one of r states. In *ordered* phylogeny problems, we know the directed sequence of transitions for each character as we move down the tree.

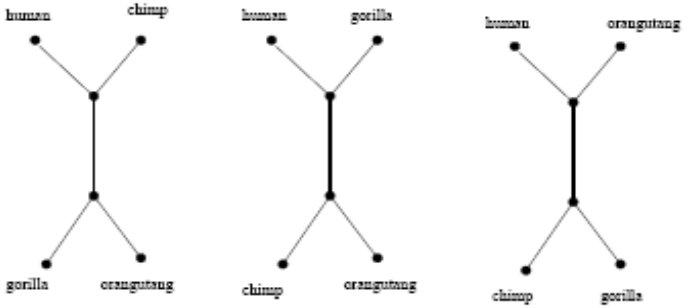
There are polynomial *ordered* perfect phylogeny algorithms for any constant r, i.e. r is in the exponent of the running time.

For unordered problems it is NP-complete for general r, but polynomial for $r = 2$.

Quartet puzzling

Another approach to reconstructing phylogenic trees is to carefully analyze small subsets of species to reconstruct their relative history, and then integrate a set of resulting trees into a consistent whole.

The smallest interesting unrooted trees contain four species, and are called *quartets*.



Quartet puzzling

There are three possible quartets on any set of species.

The set of $\binom{n}{4}$ quartets induced from any tree uniquely defines the topology of the tree.

Note that rooted triplets can be modeled as quartets if one species is ancestral.

In general, it is difficult to analyze the data to construct all possible quartets in a consistent manner. The problem of constructing the tree maximizing the number of satisfied quartets is NP-complete.

Consensus Trees

Because the various algorithms and heuristics give different trees on the same data, more evidence is needed than a single tree to define the history.

For this reason, there are suites of programs (e.g. Phylip) which contain implementations of many different tree construction algorithms and heuristics.

Thus there is a need for algorithms which find the *consensus* of a set of trees, i.e. the branch points that all (or most) of the trees share in common.

Also there are *tree compatibility* problems, where we seek the most refined tree which do not contradict evidence from any of a set of partially specified trees: