Chapter 3

Parsimony

The first approach we'll develop for inferring phylogenetic trees is called the method of Maximum Parsimony (MP), or, more informally, parsimony. The essential idea is probably the oldest of those underlying any method, and is applied both to inferring phylogenies from sequence data and from data of other types, such as morphological data.

3.1 The Parsimony Criterion

Suppose we obtained the following aligned DNA sequences for four taxa:

```
                1
       123456789012345
S1 :   AACTTGCGCATTATC
S2 :   ATCTTGCGCATCATC
S3 :   ATCTTGGGCATCATC
S4 :   AACTTGGGCATTATC
```

There are 15 different rooted topological trees that might relate these, and we want to pick 'the best' in some sense.

Note the only variations in bases are at sites 2, 7, and 12. We might interpret site 2 as evidence for a tree such as $((S_1, S_4), (S_2, S_3))$ since a single base substitution occurring on a edge leading from the root could explain the data at that site. A tree such as $((S_1, S_2), (S_3, S_4))$, in contrast, would require at least 2 substitutions at that site. Site 7 data, on the other hand, could be explained by the smallest number of changes if $((S_1, S_2), (S_3, S_4))$ were the tree. Finally, site 12 supports the same tree as site 2. These sites conflict, but it's reasonable to decide in favor of a tree reflecting the majority of them, so $((S_1, S_4), (S_2, S_3))$ is a better choice of tree than $((S_1, S_2), (S_3, S_4))$. Of course there are 13 other rooted trees we should consider as well in order to pick the best overall for this example, but we have at least developed the germ of a principled way of picking a tree: we look for a tree requiring the fewest mutations.

To generalize this viewpoint, imagine having n taxa which we wish to relate.

We collect a data sequence of *characters* which for each taxon might be in any of a finite number of *states*. For instance, if our data are aligned orthologous DNA sequences, then the characters correspond to the different sites in the sequences, and their possible states on each of the taxa are $\{\mathtt{A}, \mathtt{G}, \mathtt{C}, \mathtt{T}\}$. If our data are morphological, then we might have characters referring to wingedness, with states {straight, curved}, or some other body part with states such as {segmented, unsegmented, absent}. Other sources of characters might be genetic or biochemical features that vary among the taxa. Thus while our examples below will use DNA data, the method applies more generally.

The simplest form of the principal underlying parsimony is:

> *The best tree to infer from data is the one requiring the fewest changes in states of characters.*

Informally, then, a 'good' tree is one that could describe an evolutionary history with as little change as possible. Though there have been protracted and acrimonious debates about the philosophical justifications for this principal (for details see, for instance, [Fel04]), in some circumstances it is certainly a reasonable view. Under what circumstances it might be problematic will be considered later.

To mathematically formalize the principal we make a number of definitions:

Definition. A *character* with state set S on a set of taxa X is a function $\chi : X \to S$. If $s = |S|$, we say χ is an s-state character.

We will assume all sets are finite here. By passing to a numerical encoding of states, we could even assume $S = \{1, 2, \dots, s\}$.

Our data for the parsimony problem for a set of taxa X are then a finite sequence of characters $\mathcal{C} = \{\chi_1, \chi_2, \dots, \chi_k\}$ where χ_i is an s_i-state character on X for each i. For DNA data, the characters χ_i just refers to the ith site in the aligned sequences, k is the length of the sequences, and $s_i = 4$ for all i, with $S_i = S = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$. For morphological data, the s_i and sets S_i may vary, and the ordering of the characters would be arbitrary.

Note that we refer to a *sequence* of characters since that terminology is mathematically valid and in accord with biological usage when, say, DNA sequences are used. However, we could more generally have referred to a set or *multiset* of characters since we will not in fact use the specific ordering. Recall that multisets are unordered, but allow an element to be repeated, unlike sets. For morphological data, there is generally no natural ordering to the characters, so the multiset terminology would be more natural.

Consider a fixed phylogenetic X-tree T, either rooted or unrooted, with leaves labeled by the taxa in X. For the remainder of this chapter we'll make no distinction between the leaves of T and their labels in X.

While our data are composed of characters χ on X, to judge the number of state changes T requires, we also need to consider characters $\tilde{\chi}$ on the full vertex set $V(T)$. In accordance with standard terminology for functions, we say a character $\tilde{\chi}$ on $V(T)$ is an *extension* of χ to T if $\tilde{\chi}(x) = \chi(x)$ for all $x \in X$. That is, an extension of a character assigns the same states to the leaves as the original character, but also assigns states to internal vertices. We use $Ext_T(\chi)$ to denote the set of all extensions of χ to T, and think of its elements as representing the possible evolutionary histories that are consistent with the observation of χ.

For each edge $e = \{v, w\} \in E(T)$ and character $\tilde{\chi}$ on $V(T)$, define

$$\delta(e, \tilde{\chi}) = \begin{cases} 1 & \text{if } \tilde{\chi}(v) \neq \tilde{\chi}(w) \\ 0 & \text{otherwise} \end{cases}.$$

Viewed as a function of e, with $\tilde{\chi}$ fixed, this is the indicator function of those edges where a state change occurs in the evolutionary history $\tilde{\chi}$.

Definition. The *state-change count* for $\tilde{\chi}$ on a phylogenetic X-tree T is the number of edges on which a state change occurs for $\tilde{\chi}$:

$$c(\tilde{\chi}, T) = \sum_{e \in E(T)} \delta(e, \tilde{\chi}).$$

The *parsimony score* of an X-tree T for a character χ on X is the minimal state-change count achieved by an extension of χ:

$$ps_\chi(T) = \min_{\tilde{\chi} \in Ext_T(\chi)} c(\tilde{\chi}, T).$$

We say $\tilde{\chi}$ is a *minimal extension* of χ for T if

$$c(\tilde{\chi}, T) = ps_\chi(T).$$

These definitions are for a single character. For a sequence of characters we need the following.

Definition. The *parsimony score* of a phylogenetic X-tree T for a sequence of characters $\{\chi_1, \dots, \chi_k\}$ on a taxon set X is the sum of those for the individual characters:

$$ps_{\{\chi_i\}}(T) = \sum_{i=1}^{k} ps_{\chi_i}(T).$$

The set of *most parsimonious trees* for a sequence of characters $\{\chi_1, \dots, \chi_k\}$ is the collection of trees achieving the minimal parsimony score:

$$\{T \mid ps_{\{\chi_i\}}(T) = \min_{T'} ps_{\{\chi_i\}}(T')\}.$$

Our goal now is to find a method that will take a sequence of characters on X and find the set of most parsimonious trees for it. There are several difficulties we face. First, we need to consider all possible trees that might relate our taxa. We've already discussed enumerating these in the last chapter, so, at least in principal, we could methodically consider one tree after another. However, for even midsize n, the number of n-taxon binary phylogenetic trees is enormous. We'll return to this issue later on.

Second, for each tree we consider, to compute the parsimony score we apparently must consider all possible extensions of the data characters. Even if a character has only 2 states, since the number of internal nodes of an n-taxon rooted binary phylogenetic tree is $n-1$, the number of extensions of that character is 2^{n-1}. For DNA data the numbers becomes 4^{n-1}. This exponential growth with n means that considering each of these extensions in turn would also require an enormous amount of work, and become infeasible for a large number of taxa. Fortunately, there is a way around this computational issue, which we turn to in the next section.

3.2 The Fitch-Hartigan Algorithm

The problem of computing $ps_{\{\chi_i\}}(T)$ for a *fixed* tree T is sometimes called the *small parsimony problem*, while doing so for *all* trees T is the *large parsimony problem*. Although both problems at first appear to be computationally intensive, the small problem can in fact be solved efficiently. A simple algorithm was developed independently by Fitch and Hartigan, and proved to work as claimed by Hartigan in 1973. Although it can be easily generalized, we present it here only for binary trees. We illustrate it and motivate it by an example:

Suppose we look at a single character (in this example, a site in the aligned DNA sequences) for each of five taxa and observe states (bases) as shown:

$$\mathrm{S}_1: \texttt{A}, \quad \mathrm{S}_2: \texttt{T}, \quad \mathrm{S}_3: \texttt{T}, \quad \mathrm{S}_4: \texttt{G}, \quad \mathrm{S}_5: \texttt{A}.$$

For the leaf-labeled tree T of Figure 3.1 we wish to compute $ps_\chi(T)$.

After writing the appropriate character state at each leaf, we simply trace backwards up the tree, from the leaves to the root, drawing reasonable conclusions as to what the state of the character might be at each interior vertex, assuming the fewest possible state changes occurred. For instance, at the parent of S_1 and S_2 we could have had either an `A` or a `T`, but obviously not a `C` or a `G`, if we are to minimize state changes, and at least 1 state change must have occurred on the edges below it. We thus write the set of possibilities $\{\texttt{A}, \texttt{T}\}$ at that vertex, and have a count of 1 so far. Now, given what appears at S_3, at the most recent common ancestor of S_1, S_2, and S_3 we should have a `T` (and in fact a `T` at the parent of S_1 and S_2); no additional change is necessary, beyond the 1 we already counted. We've now labeled two interior vertices, and still have a count of 1.

We continue to trace backward through the tree, writing a state or set of possible states at each vertex. If the children of the current vertex are marked

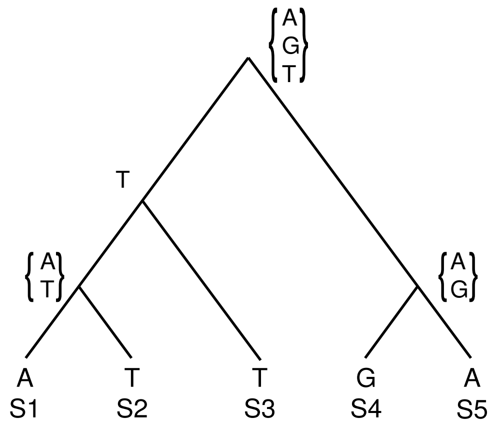

Figure 3.1: Computing the parsimony score of a tree using the Fitch-Hartigan algorithm for one character constructs sets of states at internal nodes, and yields a score of 3.

with two different states (or disjoint sets of states), we increase our state-change count by 1 and combine the two states (or the sets) to create a larger set of possible states at the parent vertex. If the two children's states agree (or the sets are not disjoint), then we label the parent vertex with that state (or the intersection of the sets). In this case no additional changes need to be counted. When all the vertices of the tree are labeled, the final value of the mutation count gives the minimum number of mutations (or state changes) needed if that tree described the evolution of the taxa. Thus the tree in Figure 3.1 would have a parsimony score of 3.

While this illustrates the algorithm, there are several points to be addressed. First, while it should seem reasonable, it is not clear that this method gives the minimum possible mutations needed for the tree. It does correctly calculate the parsimony score for the tree and character, but that must be proved. (You might also suspect the algorithm produces all minimal extensions of the character on the tree. However, it does *not*, as Exercise 9 shows. There can be assignments of states to the internal vertices that are not consistent with what this algorithm produces, yet which still achieve the same state-change count. This illustrates the importance of proving an algorithm works as one wishes; naive intuition can be misleading.)

Second, we performed the algorithm on a *rooted* tree. While we've been intentionally unclear as to whether we are considering rooted or unrooted trees with the parsimony criterion, in fact it doesn't matter, as we now show.

Theorem 7. If T^ρ is a rooted version of T, then for any character χ,

$$ps_\chi(T^\rho) = ps_\chi(T).$$

Proof. If ρ is a vertex of the tree T, then by the definition of the parsimony score this is clear.

So suppose ρ has been introduced along an edge of T, by replacing that edge with two edges meeting at ρ. Observe that any minimal extension of χ to T^ρ must have the same state at ρ as at least one of the adjacent vertices.

But then we see $ps_\chi(T^\rho) \geq ps_\chi(T)$, since when any minimal extension of χ to T^ρ is restricted to the vertices of T, its state-change count remains the same. But also $ps_\chi(T^\rho) \leq ps_\chi(T)$ since given any minimal extension of χ to T, we can further extend it to T^ρ without altering its state-change count. $\square$

Thus the parsimony score does not depend on the root location, and once we show the Fitch-Hartigan algorithm gives the parsimony score for a rooted tree, it must give the same score regardless of root location. While the details of the algorithmic steps depend on the choice of root, the final score does not.

Before we give more proofs, though, let's return to our example. Now that we've evaluated the parsimony score of the tree in Figure 3.1, let's consider another tree, in Figure 3.2, that might relate the same 1-base sequences.

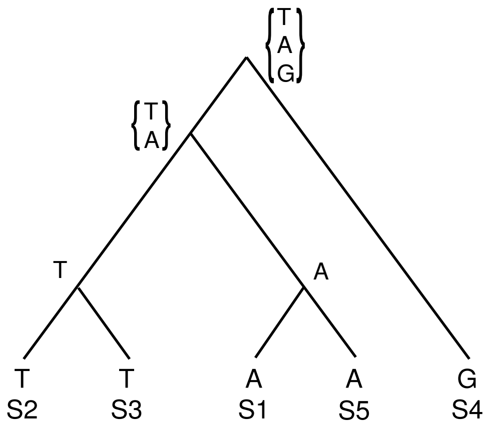

Figure 3.2: Using the same character as in Figure 3.1, the Fitch-Hartigan algorithm shows this tree is more parsimonious, with a score of 2.

Applying the method above to produce the labeling at the internal vertices, we find this tree has a parsimony score of 2; only two mutations are needed. Thus the tree in Figure 3.2 is more parsimonious than that of Figure 3.1.

To find the most parsimonious tree for these taxa we would need to consider all 15 possible topologies of unrooted trees with 5 taxa and compute the minimum number of mutations for each. Rather than methodically go through the 13 remaining trees, for this simple example we can just think about what trees are likely to have low parsimony scores. If the score is low, S1 and S5 are likely to be near one another, as are S2 and S3, but S4 might be anywhere. With this observation, its easy to come up with several more trees that have parsimony score 2 but that are topologically distinct from that of Figure 3.2.

It's also easy to see that no tree will have a parsimony score of 1, since we need at least 2 mutations to have 3 different bases among the taxa. For this example there are in fact five trees that have a parsimony score of 2, which form the set of the most parsimonious.

Here's a more succinct presentation of the Fitch-Hartigan algorithm for computing the parsimony score $ps_\chi(T)$ of a binary tree T, for a single character χ on X with state space S.

Algorithm (Fitch-Hartigan).

1. If T is unrooted, arbitrarily introduce a root, to get a rooted binary tree T^ρ.

2. Assign to each vertex $v \in V(T^\rho)$ a pair (U, m) where $U \subseteq S$ and $m \in \mathbb{Z}^{\geq 0}$ as follows:

   (a) To each leaf $x \in X$, assign the pair $(\{\chi(x)\}, 0)$.

   (b) If the two children of v have been assigned pairs (U_1, m_1) and (U_2, m_2), then assign to v the pair

   $$(U, m) = \begin{cases} (U_1 \cup U_2,\ m_1 + m_2 + 1), & \text{if } U_1 \cap U_2 = \emptyset, \\ (U_1 \cap U_2,\ m_1 + m_2), & \text{otherwise.} \end{cases}$$

   Repeat until all vertices have been assigned pairs.

3. If the pair (U, m) has been assigned to ρ, then $ps_\chi(T) = m$.

With real data, we of course need to count the number of state changes required for a tree among *all* characters. Since the score for a sequence of characters is the sum of scores for each character, this can be done in the same manner as above, just treating each character in parallel. An example is given in Figure 3.3.

Proceeding up the tree beginning with the two taxa sequences ATC and ACC on the far left, we see we don't need mutations in either the first or third site, but do in the second. Thus the mutation count is now 1, and the ancestor vertex is labeled as shown. At the vertex where the edge from the third taxa joins, we find the first site needs a mutation, the second does not, and the third does. This increases the mutation count by 2, to give us 3 so far. Finally, at the root, we discover we need a mutation only in the second site, for a final parsimony score of 4.

While this is not hard to do by hand with only a few sites, as more sites are considered it quickly becomes a job best done by a computer.

We now prove the Fitch-Hartigan algorithm actually produces the correct parsimony score for a rooted tree. For simplicity, we consider only binary trees. The key idea is to prove something slightly stronger, as expressed in the last sentence of the following.

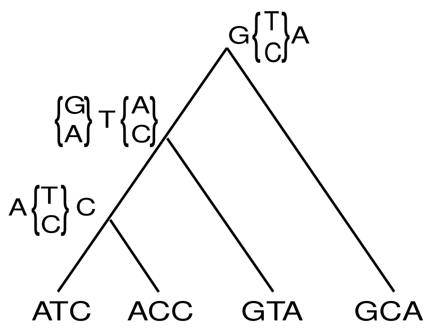

Figure 3.3: For multiple characters, the Fitch-Hartigan algorithm can be applied in parallel, yielding a score of 4 in this example.

Theorem 8. Let χ be a character on X, and T^ρ a rooted binary X-tree. Then the Fitch-Hartigan algorithm computes $ps_\chi(T)$. Furthermore, the set of states it assigns to the root ρ is exactly the set of states that occur at ρ in the minimal extensions $\tilde{\chi}$ of χ.

Proof. We proceed by induction on the size of the set X. The $|X| = 2$ case is clear.

For general $|X|$, let $v_1, v_2 \in V(T^\rho)$ be the children of ρ. Let T_i be the subtree, rooted at v_i, of descendants of v_i, and X_i denote the labels on the leaves of T_i, so that $X = X_1 \cup X_2$ is a disjoint union. Let $\chi_i = \chi|_{X_i}$. By the inductive hypothesis, the Fitch-Hartigan algorithm assigns to v_i the pair (U_i, m_i) where U_i is exactly the set of states that occur at v_i in minimal extensions of χ_i on T_i, and m_i is $ps_{\chi_i}(T_i)$.

Suppose $\tilde{\chi}$ is a minimal extension of χ to T, and let $\tilde{\chi}_i = \tilde{\chi}|_{V(T_i)}$. Then (see Exercise 7a) minimality ensures one of the following must hold:

(1) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$,

(2) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$, or

(3) $\tilde{\chi}(\rho) = \tilde{\chi}(v_2) \neq \tilde{\chi}(v_1)$.

Consider first case (2). Then $\tilde{\chi}_2$ must be a minimal extension of χ_2, for otherwise we could contradict the minimality of $\tilde{\chi}$ by defining a character on T that agreed with $\tilde{\chi}$ on $V(T_1) \cup \{\rho\}$ and agreed with a minimal extension of χ_2 to T_2 on $V(T_2)$.

We similarly see that $\tilde{\chi}_1$ must be a minimal extension of χ_1 by applying the same argument to the minimal extension $\tilde{\tilde{\chi}}$ of χ defined by

$$\tilde{\tilde{\chi}}(v) = \begin{cases} \tilde{\chi}(v) & \text{if } v \neq \rho, \\ \tilde{\chi}(v_2) & \text{if } v = \rho. \end{cases}$$

Thus in this case $\tilde{\chi}_1, \tilde{\chi}_2$ are both minimal extensions of χ_1, χ_2, respectively, while $\tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$. Using the inductive hypothesis, this shows $ps_\chi(T) = m_1 + m_2 + 1$. It also shows there do not exist any minimal extensions of χ_i on the T_i which agree on v_1 and v_2, since the existence of such would allow us to construct an extension of χ on T with a state-change count of $m_1 + m_2$, contradicting the minimality of $\tilde{\chi}$. Thus the U_i are disjoint, and $\tilde{\chi}(\rho) \in U_1 \cup U_2$. Finally, we note that for each choice of an element of $U_1 \cup U_2$ we can use minimal extensions of the χ_i on T_i to define a minimal extension of χ taking on the specified choice of state at ρ. This completes the proof in case (2). Case (3) is essentially the same.

In case (1), we cannot conclude that both $\tilde{\chi}_i$ are minimal extensions of the χ_i, but only that at least one must be (see Exercise 7b). Assume then that $\tilde{\chi}_1$ is minimal. We consider two subcases, according to whether $\tilde{\chi}_2$ is (a) minimal, or (b) not minimal.

In subcase (1a), we have that both $\tilde{\chi}_i$ are minimal, and $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$. Thus $ps_\chi(T) = m_1 + m_2$, the U_i are not disjoint, and $\tilde{\chi}(\rho) \in U_1 \cap U_2$. To see that for every $s \in U_1 \cap U_2$ there is a minimal extension with $\tilde{\chi}(\rho) = s$, we choose any minimal extensions χ_1, χ_2 with $\chi_1(v_1) = \chi_2(v_2) = s$ and define $\tilde{\chi}$ to agree with them on the subtrees and have $\tilde{\chi}(\rho) = s$. Such a $\tilde{\chi}$ achieves the minimal score.

In subcase (1b), pick any minimal extension $\tilde{\tilde{\chi}}_2$ of χ_2. Define a new character $\tilde{\tilde{\chi}}$ on $V(T)$ to agree with $\tilde{\chi}$ on $V(T_1) \cup \{\rho\}$, and agree with $\tilde{\tilde{\chi}}_2$ on $V(T_2)$. Note that the state-change count for $\tilde{\tilde{\chi}}$ cannot be greater than that of $\tilde{\chi}$, since the number of state changes on edges in T_2 has been decreased, at the expense of only one additional state change on the edge $\{\rho, v_2\}$. Since it also cannot have a lower state-change count by the minimality of $\tilde{\chi}$, $\tilde{\tilde{\chi}}$ is a minimal extension of χ, with both $\tilde{\tilde{\chi}}_1$ and $\tilde{\tilde{\chi}}_2$ minimal on the subtrees. Thus $ps_\chi(T) = c(\tilde{\tilde{\chi}}, T) = m_1 + m_2 + 1$, and $\tilde{\chi}(\rho) = \tilde{\tilde{\chi}}(\rho) \in U_1 \cup U_2$. From $ps_\chi(T) = m_1 + m_2 + 1$, it is straightforward to see that the U_i must be disjoint. Finally, that a minimal extension of χ exists with any element in $U_1 \cup U_2$ as its value at ρ is as in case (2). □

3.3 Informative Characters

We can save some computational effort in finding parsimonious trees if we observe that some characters do not affect the parsimony scores of trees in ways that affect our judgment of which is optimal.

The obvious case is a character χ that assigns the same state to all taxa, *i.e.*, a constant character. This character extends to a constant character on $V(T)$ for any T, and hence $ps_\chi(T) = 0$ for all T. Thus we can simply discard the character from a sequence of characters, and not change parsimony scores.

A less obvious case is a character χ that assigns the same state to all taxa but a few, assigning different states to each of these few taxa. For example, suppose a character assigned the state `A` to one taxon, `G` to another, and `C` to all others. Then whatever tree is considered will certainly require at least 2 state changes, since 3 states are observed. On the other hand, regardless of the tree, the extension of the character that assigns a `C` to all internal nodes would achieve a state-change count of 2. Thus this character will have a parsimony score of 2 on every tree, and its inclusion in a data set of characters will only increase the parsimony score of every tree by the same amount. It therefore will have no effect on our choice of the best tree under the parsimony criterion.

To formalize this reasoning, suppose $|\chi^{-1}(i)| > 1$ for at most one state i. In this case, let j denote a state with $|\chi^{-1}(j)|$ maximal, and $l = |\chi(X)|$. Then for any T we can extend χ to T by assigning to each internal vertex the state j. This shows $ps_\chi(T) \leq l - 1$. Since $ps_\chi(T) \geq l - 1$ for any χ and T (see Exercise 8), this means $ps_\chi(T) = l - 1$ for all trees T. While the appearance of such a χ among a sequence of characters *does* affect the parsimony score of all trees, it simply inflates them all by the same amount, and so *does not* affect the selection of the most parsimonious trees.

This leads to the idea of an *parsimony-informative character*.

Definition. A character on X is *(parsimony-)informative* if it assigns to the taxa at least two different states at least twice each. That is, χ is informative if, and only if,

$$\left|\left\{i \in S \ : \ \left|\chi^{-1}(i)\right| > 1\right\}\right| > 1.$$

Before applying the Fitch-Hartigan parsimony algorithm, we can eliminate all non-informative characters from our data since they will not affect the choice of most parsimonious trees. For DNA sequence data, many characters are likely to be non-informative, since identical, or nearly-identical sites are needed to identify and align orthologous sequences. In the examples above, you'll notice only informative sites have been used.

There are several explicit warnings we should make concerning the notion of informative sites. First, the concept applies *only* when one is using the parsimony criterion for tree selection. Under other methods, deletion of these sites will lead to biased results — they contain useful information under other methods.

Second, even under parsimony the non-informative sites only have no information for selecting the *topological* tree. After finding the most parsimonious topological tree, sometimes lengths are assigned to edges as a measure of how many state changes had to occur on that edge. To do this, all minimal extensions of the characters are considered (see Section 3.6), and for each edge an average is taken of the counts associated to it by each extension. The resulting metric tree does depend on all characters except the ones that are identical for all taxa.

Additional 'pre-processing' of the data can reduce the computations a bit more. For instance, the particular states of a character at the leaves of a tree do not matter, as long as we understand which are different. For instance on a 5-taxon tree, a character that has states `A`,`C`,`C`,`G`,`A` at leaves 1-5 will produce the same parsimony score as a character with states `G`,`A`,`A`,`T`,`G` at those leaves, since both are instances of a pattern of the form *xyyzx*. By counting the number of occurrences of each such pattern form in the data, parsimony scores can be found by only performing the Fitch-Hartigan algorithm on a single representative of each.

3.4 Complexity

If an algorithm is to be performed on a large data set, it is important to understand how much work this will require. Even if a computer is used (as of course is always the case), if the amount of work is too great, runtimes can be unacceptably slow for an algorithm to be useful. For instance, if our approach to the small parsimony problem was not the Fitch-Hartigan algorithm, but rather the more naive and cumbersome approach of listing all possible extensions of our characters, computing a score for each, and then picking the smallest, we could never expect to have software capable of analyzing data sets with a large number of taxa in an acceptable amount of time. We've already seen that the number of such extensions is exponential in the number of taxa, and thus impossibly large once there are many taxa.

To see that the Fitch-Hartigan algorithm is better than this, we have to quantify the amount of work in it. To do this, first consider a single character on an n-taxon set, with s possible states. A measure of the work to perform a basic step of the process (*i.e.*, to compare the sets of states at two child nodes, determine the set for the parent, and possibly increase the state-change count) is Cs, since we will have to consider each possible state in turn, and do a fixed amount of work C for each. Since a rooted binary n-leaf tree has $n-1$ internal vertices, we will perform this work $n-1$ times, for a total of $Cs(n-1)$. If there are k characters, then the work must be done for each, for a total of $Csk(n-1) < Cskn$. Thus the amount of work grows only linearly in the number of taxa, a vast improvement over the exponential growth of the naive approach.

Although we have not given a value for C, there is certainly such a constant (which might be measured in units of time, or computer operations). In standard jargon we would say the Fitch-Hartigan algorithm has runtime $\mathcal{O}(skn)$ (read "big-O of skn"), to indicate the runtime is less than some unspecified constant times the given product. In general, algorithms with polynomial runtimes are considered feasible, and those with exponential or worse are problematic. The linear runtime here is excellent. (If the unspecified constant were huge, that might also be a problem, but at least for the Fitch-Hartigan algorithm it should be clear that it is quite small.) Thus the algorithm gives a very efficient way of solving the small parsimony problem.

However, this is only for one tree T. For the large problem, we must compare parsimony scores for all unrooted binary trees, and there are $(2n-5)!!$ of them to consider. Searching through them all would be horribly slow. A natural question asks if this slowness is unavoidable.

The answer to this last question appears to be 'yes.' More technically, finding a most parsimonious tree is an NP-hard problem[1], and so we do not expect any algorithm will be found that *guarantees* we have found all (or even one) optimal trees when n is large. (That no NP-hard problem can be solved in polynomial time has not yet been proved, but is widely believed. It is one of the most famous open mathematical questions.)

In practice, there are heuristic search methods that seem to work well and reasonably quickly. However, they do this by not considering all possible trees. In some circumstances, they can rule out a class of trees as not possibly containing the most parsimonious. But more often, after finding a reasonably good tree they spend most of their time focusing on only searching among trees that are "similar". As a result, only when the number of taxa is small can we be sure we have really found the most parsimonious trees. We will delay a discussion of methods of exploring similar trees in a heuristic search until a later chapter.

3.5 Weighted Parsimony

A natural generalization of the basic idea of parsimony is to introduce different penalties for different state changes in computing parsimony scores. For instance, in DNA sequence data it is often clear that transitions and transversions occur at different rates. If transversions are more rare, they may be less prone to occurring multiple times at the same site than transitions, and thus preserve more phylogenetic signal. We might then choose to give them a higher weight in calculating parsimony scores, to take advantage of this. We now develop an algorithm for computing the weighted parsimony scores that would result from such a scheme.

The Fitch-Hartigan algorithm for computing unweighted parsimony scores is an example of a *dynamic programming* algorithm, a rather common algorithmic approach to solving problems. Roughly this means the algorithm proceeds by finding optimal solutions to smaller instances of the problem (on subtrees) in order to use those solutions to get an optimal solution to the problem we care about.

The dynamic programming approach was used by Sankoff to develop an algorithm for computing weighted scores. It is very similar to the Fitch-Hartigan algorithm, but requires more bookkeeping as we work our way through the tree.

Suppose χ is an s-state character on X, and fix an $s \times s$ matrix $W = (w_{ij})$ of weights. Here w_{ij} is the cost we wish to impose for a state change from state i to state j in the descent from the root. While it is natural to assume $w_{ii} = 0$ and $w_{ij} \geq 0$ for $i \neq j$, it is not required that we do so.

For a rooted X-tree T^ρ, the weighted parsimony score is defined as follows:

[1]This is due to Foulds and Graham (1983). Ron Graham was sent to Fairbanks while in the U.S. AIr Force in the 1950s, then earned his undergraduate degree at UAF. He was later awarded an honorary doctorate from UAF.

Definition. For any extension $\tilde{\chi}$ of χ on T^ρ, the *cost* of $\tilde{\chi}$ is

$$c_W(\tilde{\chi}, T^\rho) = \sum_{e \in E(T)} w_{\tilde{\chi}(t_e)\tilde{\chi}(h_e)}.$$

Here $t_e, h_e \in V(T)$ denote the tail and head vertices of the edge e directed away from the root.

Definition. The *weighted parsimony score* of T^ρ is

$$ps_{\chi,W}(T^\rho) = \min_{\tilde{\chi} \in Ext_{T^\rho}(\chi)} c_W(\tilde{\chi}, T^\rho).$$

Rather then begin with an example, we'll formally state an algorithm for computing weighted parsimony scores first. Again, we only treat binary trees, since generalizing to arbitrary trees is straightforward. Suppose X, T^ρ, and χ are given.

Algorithm.

1. Assign to each leaf $x \in X$ of T^ρ an s-element vector $\mathbf{c}$, where

$$c_i = \begin{cases} 0 & \text{if } \chi(x) = i, \\ \infty & \text{otherwise .} \end{cases}$$

2. If the two children of $v \in V(T)$ have been assigned vectors $\mathbf{a}$ and $\mathbf{b}$, then assign to v the vector $\mathbf{c}$ with

$$c_i = \min_{j \in S}(w_{ij} + a_j) + \min_{k \in S}(w_{ik} + b_k).$$

   Repeat until all vertices have been assigned vectors.

3. If the vector $\mathbf{c}$ has been assigned to ρ, output $m = \min_i(c_i)$.

Figure 3.4 shows an example of the algorithm, where transitions have been given weight 1, transversions weight 2, and no substitution weight 0. The final score for the tree is thus found to be 3. (While the character in this example is uninformative for *un*weighted parsimony, that does not mean it must be uninformative if weights are used. See Exercise 19.)

We'll omit a formal proof that Sankoff's algorithm gives the correct weighted parsimony score, as it is similar to that for the Fitch-Hartigan algorithm.

Note also that allowing weights to be assigned to state changes may have made the parsimony score of a tree dependent on the root location. While this may be desirable for some purposes, it means we will have to search through all rooted trees rather than just unrooted ones. If we restrict the allowable choices of weights for state changes, however, we can preserve the independence of the parsimony score from the root location. We leave determining how this is done as Exercise 16.

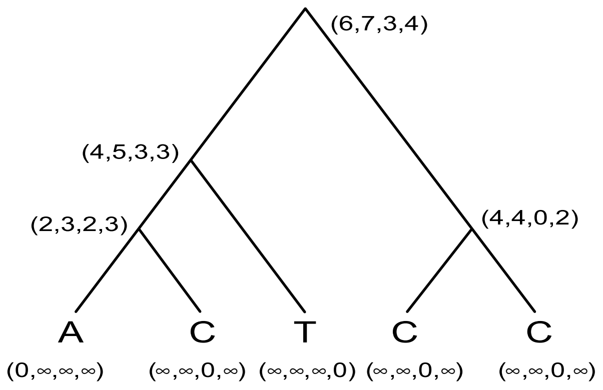

Figure 3.4: The Sankoff algorithm for computing weighted parsimony scores, requires that scoring vectors be computed at each internal node. In this example, transitions are weighted 1 and transversions weighted 2, with the base order A, G, C, T used for scoring vectors.

3.6 Recovering Minimal Extensions

In addition to the parsimony score of a tree, we may actually want to know what minimal extensions of χ achieve that score. Thinking of minimal extensions as likely evolutionary histories, we might want to know for each such history which state changes occurred on which edges of a tree. Across many characters, this would allow us to identify on which edges we had many state changes and which had few, thus indicating something about either the length of time those edges represent, or whether circumstances were highly favorable to state changes during those periods. Of course the various minimal extensions may indicate state changes along different edges, so at best we will be able to assign a range to the number of changes (or the total cost for weighted parsimony) along each edge, or the average over all minimal extensions.

Although we'll outline how to find all minimal extensions only in the setting of the weighted parsimony algorithm of Sankoff, there is also a more streamlined variation for the unweighted case that builds on the Fitch-Hartigan algorithm.

Suppose then that we have computed the weighted parsimony score of a character χ on a tree T^ρ by the algorithm above. If $\mathbf{c}$ is the vector assigned to ρ, then for each i with c_i minimal, there will be a minimal extension of χ taking on state i at ρ.

Considering only one such minimal c_i, let v_1 and v_2 be the children of ρ, with assigned vectors $\mathbf{a}, \mathbf{b}$. Then for each choice of state j at v_1 and state k at v_2 with

$$c_i = w_{ij} + a_j + w_{ik} + b_k,$$

there will be a minimal extension of χ taking on the states i, j, k at ρ, v_1, v_2, respectively. We simply continue down the tree in this way, to eventually produce all minimal extensions.

3.7 Further Issues

There are a number of issues with parsimony that we won't discuss in detail, but that we leave for you to think about:

1. If several trees tie for most parsimonious (as they often do), what should we do? There are various approaches to defining *consensus trees* that attempt to summarize the features of several trees by a single tree. We will touch on this in Chapter 4

2. What about trees that are 'close' to most parsimonious? Are we really committed enough to the parsimony principle to feel that being only a few state changes off from most parsimonious means a tree should not be considered as biologically reasonable?

3. With more than a handful of taxa, there are many possible binary trees to consider that might relate them. If there are too many to search through to be sure of finding the most parsimonious, what heuristic search methods are likely to do well in practice?

4. If weighted parsimony is to be used, how should weights be assigned? Since assigning weights can affect which trees we view as 'best,' how can we objectively choose?

5. We can also choose to weight various characters differently in computing the parsimony score of a tree. This is different than assigning weights to different state changes for a particular character: For instance we might have morphological characters for body size and for wing shape of insects, and decide that wing shape changes are more significant indicators of evolutionary relationships than body size ones, so they should count twice as much in the parsimony score. More generally, it might be reasonable to assign higher weights to characters representing features of an organism that change less often, and lower ones to characters that vary more freely. But if this is done, how do we justify the precise choices of those weights? (Conversely, if this is not done, how do we justify giving all characters equal weights?)

6. A final issue, which has been the subject of much controversy among biologists, concerns the philosophical underpinning of parsimony. There are strong proponents of parsimony who believe it is the *only* valid way of inferring phylogenies. There are others who may prefer other methods but admit parsimony is a reasonable approach under some circumstances. In Chapter 11 we will see that some reasonable probabilistic models of mutation processes can sometimes lead to data that will cause parsimony to infer an incorrect tree.

We cannot yet explore this issue fully, but will give a hint as to part of the problem. If, say, we are discussing DNA mutation, then along an edge of a tree that represents a long period of time a single site may mutate several times, for instance A $\to$ C $\to$ G, or A $\to$ C $\to$ A. Thus while two state changes occurred, either one or none is observed if we only consider the ends of the edge. The parsimony principle, however, rejects the possibility of multiple changes as a preferred explanation.

As long as state changes are rare on all edges of a tree, parsimony is a quite reasonable approach to inferring a phylogeny. It is when state changes are less rare that potential problems arise. For characters describing morphology, or other larger-scale observations, we often expect few, if any, hidden mutations. For sequence data, the situation is less clear and depends on the data set being examined.

3.8 Exercises

1. a. Using the Fitch-Hartigan algorithm, compute the minimum number of base changes needed for the trees in Figure 3.5.

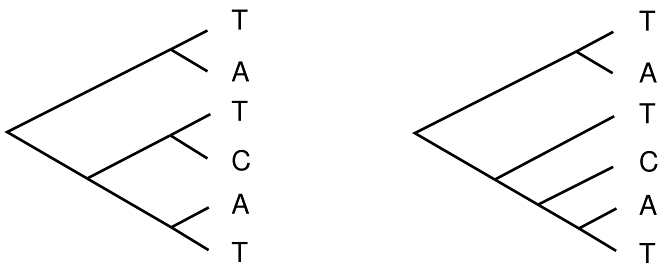

Figure 3.5: Trees for Problem 1

b. Give at least three trees that tie for most parsimonious for the one-base sequences used in part (a).

c. For trees tracing evolution at only one DNA site as in (a) and (b), why can we always find a tree requiring no more than 3 substitutions no matter how many taxa are present?

2. a. Find the parsimony score of the trees in Figure 3.6. Only informative sites in DNA sequences are shown.

b. Draw the third possible (unrooted) topological tree relating these sequences and find its parsimony score. Which of the three trees is most parsimonious?

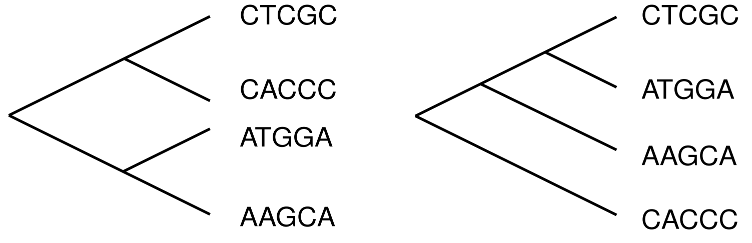

Figure 3.6: Trees for Problem 2

3. What changes are necessary to the Fitch-Hartigan algorithm if a tree has a polytomy? Give an example of how you would handle a vertex with 3 children.

4. Consider the following sequences from four taxa.

```
              1
     123456789012345
S1:  AATCGCTGCTCGACC
S2:  AAATGCTACTGGACC
S3:  AAACGTTACTGGAGC
S4:  AATCGTGGCTCGATC
```

a. Which sites are informative?

b. Use the informative sites to determine parsimony scores for the three possible unrooted trees relating the taxa. Which is the most parsimonious?

c. If S_4 is known to be an outgroup, use your answer to (b) to give a rooted tree relating S_1, S_2, and S_3.

5. Though non-informative sites in DNA do not affect which tree is judged to be the most parsimonious, they do affect the parsimony score. Explain why, if P_{all} and P_{info} are the parsimony scores for a tree using all sites and just informative sites, then

$$P_{\text{all}} = P_{\text{info}} + n_1 + 2n_2 + 3n_3,$$

where, for $i = 1, 2, 3$, by n_i we denote the number of sites with all taxa in agreement except for i taxa which are all different . (Notice that while P_{all} and P_{info} may be different for different topologies, $n_1 + 2n_2 + 3n_3$ does not depend on the topology.)

6. Show that if χ is an informative character on a set X of n taxa (so $n \geq 4$), then there exist two phylogenetic X-trees T_1, T_2 with $ps_\chi(T_1) \neq ps_\chi(T_2)$.

7. Complete the proof of Theorem 8 by doing the following:

(a) Suppose $\tilde{\chi}$ is a minimal extension of χ to T, and let $\tilde{\chi}_i = \tilde{\chi}|_{V(T_i)}$, with the T_i as defined in the proof. Explain why minimality ensures one of the following must hold: (1) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) = \tilde{\chi}(v_2)$, (2) $\tilde{\chi}(\rho) = \tilde{\chi}(v_1) \neq \tilde{\chi}(v_2)$, or (3) $\tilde{\chi}(\rho) = \tilde{\chi}(v_2) \neq \tilde{\chi}(v_1)$.

(b) Explain why in case (1) at least one of the $\tilde{\chi}_i$ is a minimal extension of χ_i. Give an example to show both need not be minimal. (You can do this with a 4-leaf tree.)

8. Suppose χ is any character on X, and let $l = |\chi(X)|$. Explain why the lower bound $ps_T(\chi) \geq l - 1$ holds for any phylogenetic X-tree T.

9. For the character and first tree in Figure 3.7, calculate the parsimony score, labeling the interior vertices according to the Fitch-Hartigan algorithm. Then show that the second tree requires exactly the same number of base changes, even though it is not consistent with the way you labeled the interior vertices on the first tree. (The moral of this problem is that naively interpreting the Fitch-Hartigan algorithm will not produce all minimal extensions of a character.)

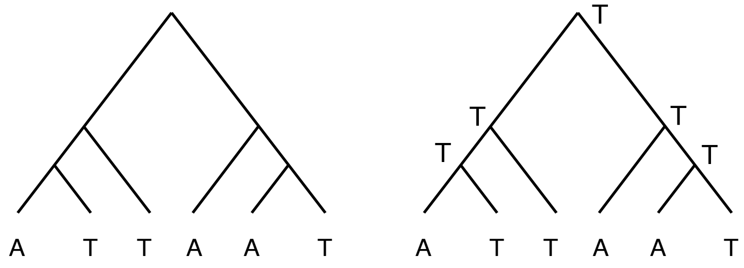

Figure 3.7: Trees for Problem 9

10. If characters are given for 3 terminal taxa, there can be no informative ones. Explain why this is the case, and why it doesn't matter.

11. The bases at a particular site in aligned DNA sequences from different taxa form a *pattern*. For instance, in comparing $n = 5$ sequences at a site, the pattern (ATTGA) means A appears at that site in the first taxon's sequence, T in the second's, T in the third's, G in the fourth's, and A in the fifth's.

a. Explain why in comparing sequences for n taxa, there are 4^n possible patterns that might appear.

b. Some patterns are not informative, such as the 4 patterns showing the same base in all sequences. Explain why there are $(4)(3)n$ non-informative patterns that have all sequences but one in agreement.

c. How many patterns are non-informative because 2 bases each appear once, and all the others agree?

d. How many patterns are non-informative because 3 bases each appear once, and all others agree?

e. Combine your answers to calculate the number of informative patterns for n taxa. For large n, are most patterns informative?

12. A computer program that computes parsimony scores might operate as follows: First compare sequences and count the number of sites f_i for each informative pattern p_i that appears (e.g., for 4 taxa, we might have $p_1 =$ `AAAA`, $p_2 =$ `AAGG`$, \dots$). Then, for a given tree T, calculate the parsimony score $ps_{p_i}(T)$ for each of these patterns. Finally, use this information to compute the parsimony score for the tree using the entire sequences. What formula is needed to do this final step? In other words, give the parsimony score of the tree in terms of the f_i and $ps_{p_i}(T)$.

13. Parsimony scores can be calculated even more efficiently by using the fact that several different patterns always give the same score. For instance, in relating four taxa, the patterns `ATTA` and `CAAC` will have the same score.

    a. Using this observation, for 4 taxa how many different informative patterns must be considered to know the parsimony score for all?

    b. Repeat part (a) for 5 taxa.

14. Use the Sankoff algorithm to computed the weighted parsimony score, with 1:2 weights for transitions:transversions, for the following aligned sequences on the tree which shows A and B most closely related.

A:	`TGA`
B:	`TAT`
C:	`TGT`
D:	`TAC`

15. For DNA data, what weight matrix makes weighted parsimony the same as unweighted parsimony? Using it, perform the Sankoff algorithm on the tree and character in Figure 3.1, and see that you recover the same score the Fitch-Hartigan algorithm produced, and the same states at the root.

16. What natural condition on the weight matrix ensures that weighted parsimony will always give the same score to any character extension regardless of the root location of the tree?

17. How would you modify the Sankoff algorithm for a tree with a polytomy?

18. Create an example of 4 sequences where unweighted parsimony leads to a different optimal tree than weighted parsimony with the transversion weight twice that of transitions.

19. When using weighted parsimony, one must be careful with the notion of an uninformative character. Consider a scheme in which transitions and transversions are weighted as 1 and 2, as in the example in Figure 3.4.

a. Explain why for the character in Figure 3.4 under this weighting every tree will produce a score of no more than 3. (Hint: assign all internal nodes the same state to achieve this score.)

b. Explain why for the character in Figure 3.4 under this weighting no tree can achieve a score lower than 3, and thus the character is uninformative under the weighted scheme.

c. Change the character so the taxon with state `T` instead has state `G`. Then find two trees on which this new character has different parsimony scores. This shows that the character is informative under the weighted scheme, though it would be uninformative under standard (unweighted) parsimony.

20. Explain why the weighted parsimony algorithm for one s-state character on a set of taxa X will involve $\mathcal{O}(s^2|X|)$ steps.

21. When applying weighted parsimony to sequence data such as for DNA, it is reasonable to require the weight matrix satisfy the condition

$$w_{ij} + w_{jk} \geq w_{ik}, \text{ for all } i, j, k.$$

Explain why, and how this relates to the possibility of multiple mutations along an edge.

22. How would you modify the Sankoff algorithm for weighted parsimony to deal with missing information on a character's value for some of the taxa?