CS5238 Combinatorial methods in bioinformatics 2002/2003 Semester 1

Lecture 7: Phylogenetic Trees Reconstruction

Date-September-27,2002

Lecturer: Wing-Kin Sung Scribe: Hemendra S. Negi,Li Haiquan, Xu Zhou, Pavandip S. Wasan,Koot Poon Wei

7.1 Introduction

Deoxyribonucleic Acid or DNA is commonly known to be responsible for encoding the information of life. Through sexual reproduction, DNA is passed on as hereditary material to offspring. During the replication of genes to be passed on to the offspring, sometimes 'mistakes' might occur. These mistakes cause the DNA, instead of being identical to the parent DNA, to change a little(one or more of the bases in the strand might have been changed). Such 'mistakes' are known as mutations. Through many generations of reproductions, with mutations going on between every generation, bringing about little changes each time in the offspring, different species emerge(or evolve). This is where phylogenetic comes in. Phylogenetic is the study of the genetic relationship among different species.

7.1.1 Mitochondrial DNA and Inheritance

The evolution of modern man is one of the interesting topics in phylogenetics. Before going further into details on the evolution of modern man, a particular aspect which has proven to be significant in this study will be discussed - mitochondrial DNA(mtDNA). mtDNA is located in the mitochondria of the cell. The mitochondria are organelles located outside the nucleus in the cytoplasm of the cell. These organelles are responsible for energy transfer and are basically the 'powerhouses' of the cell. There are about 1,700 mitochondria in every human cell, each includes an identical circular DNA of about 16,000 base pairs long containing 37 genes. This form of DNA is in short strands and therefore does not mutate very quickly - it is relatively stable and can be compared across several generations. Whenever an egg cell is fertilized, nuclear chromosomes from a sperm cell enter the egg and combine with the egg's nuclear DNA, producing a mixture of both parents' genetic code. The mtDNA from the sperm cell, however, is left behind, outside of the egg cell. So the fertilized egg contains a mixture of the father and mother's nuclear DNA and an exact copy of the mother's mtDNA,

but none of the father's mtDNA. The result is that mtDNA is passed on only along the maternal line. This means that the mtDNA in the cells of a person's body are copies of his or her mother's mtDNA, and all the mother's mtDNA is a copy of her mother's, and so on.

7.1.2 The Constant Molecular Clock

Even though everyone in the Earth today has inherited his or her mtDNA from one person who lived long ago, our mtDNA is not exactly alike. Random mutations have altered the genetic code over the millennia. But these mutations are organized, in a way. For example, let's say that 10,000 years after the most recent common ancestor, one of the mtDNA branches experienced a mutation. From that point on, that line of mtDNA would include that alteration.

Another branch might experience a mutation in a different location. This alteration would also be passed on. By looking at the similarities and differences of the mtDNAs of the nowadays individuals, researchers try to reconstruct where the branching took place. From an original 1987 Nature article, the three authors (Rebecca Cann, Mark Stoneking, and Allan Wilson) looked at the mtDNA of 147 people from continents around the world (though for Africans, they relied on African Americans). Later, with the help of a computer program, they put together a sort of family tree, grouping those with the most similar DNA together, then grouping the groups, and then grouping the groups of groups. The tree they ended up showed that one of the two primary branches consisted only of African mtDNA and that the other branch consisted of mtDNA from all over the world, including Africa. From this, they inferred that the most recent common mtDNA ancestor was an African woman. This study was done under the assumption of a constant molecular clock. Such a theory supports the idea that rates of change in genes and proteins are constant over time. It claims that in any given DNA sequence, mutations accumulate at an approximately constant rate as long as the DNA sequence retains its original functions. The difference between the sequences of a DNA segment (or protein) in two species would then be proportional to the time since the species diverged from a common ancestor (coalescence time). This time may be measured in arbitrary units and then it can be calibrated in millions of years for any given gene if the fossil record of that species happens to be rich. Moreover, such a theory does not hold in general.

Molecular clocks do not behave metronomically, i.e., neutral mutations do not yield literally constant rates of molecular change but are expected to yield constant average rates of change over a long period of time (known as 'stochastically constant' rates). In general, rRNA evolves slowly and mtDNA rapidly. Fast mutating sequences cannot be used to go back in evolutionary time because the mutations effectively randomize the sequences. It is also possible that reverse mutations will occur and the comparisons will be very difficult. These facts along with further research have that Wilson's clock is not accurate and that the anal-

ysis only support the common ancestor of modern man appear from 100,000 to 1 million years ago.

7.1.3 Phylogeny

In Wilson's project, the tree they built is known as phylogeny. Phylogeny is defined as the reconstruction of the evolutionary history of a set of species. It is usually represented by a leaf-labeled tree where the internal nodes refer the hypothetical ancestors and the leaves are labeled by the species. Two species that look alike will be represented as neighboring external branches and will be joined to a common parent branch. The objective of phylogenetic analysis is to analyze all the branching relationships in a tree and their respective branch lengths. As an example, we can look at the phylogeny of lizards as illustrated in Figure 7.1.

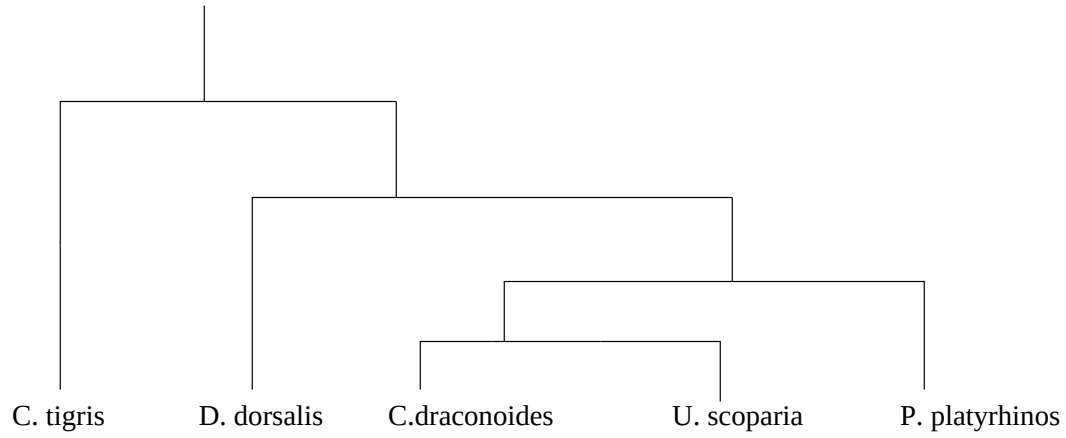


Figure 7.1: Phylogeny of lizards

Let us observe the phylogenetic tree in Figure 7.1, also known as a cladogram or a dendrogram. It is a tree of several life forms and their relations. In the figure, time is the vertical dimension with the current time at the bottom and earlier times above it. There are five extant species (species currently living) with their respective names stated. The lines above the extant species represent the same species, just in the past. When two lines converge to a point, that should be interpreted as the point when the two species diverges from a common ancestral species, the point being the common ancestral species. And so it goes until eventually, some time in the past, all the species derived from just one species, the one displayed as the top point.

When performing analysis of phylogenetic trees, one has to understand that the genomes of most organisms have a complex origin. Through reproduction of species, some parts of the genome are passed on by a vertical descent. Other parts may have arisen by horizontal transfer of genetic material between species

through a virus, DNA transformation, symbiosis(living together), or some other horizontal transfer mechanism.

7.1.4 Applications of Phylogeny

Aside from the more obvious applications of understanding the history of life and analyzing rapidly mutating viruses such as HIV, phylogeny has several other uses. One example is multiple sequence alignment. Most multiple sequence alignment programs used in practice rely on a phylogenetic tree in order to speed up the computation. In addition to that, phylogeny also helps in the prediction of the structure of proteins and RNA, helps to explain and predict gene expression, ligand structure for drug design and helps to design enhanced organisms such as rice and wheat. The most significant of these applications is the use of phylogenetic analysis in sequence alignment. When the sequences of two nucleic acid or protein molecules found in two different organisms are similar, they are likely to have been derived from a common ancestor sequence. Sequence alignments reveal conserved, and non-conserved regions between evolutionarily related sequences. When an evolutionary relationship is seen to exist between any two sequences, they are known as homologous. Therefore, one can say that sequence homology is a function of similarity. The analysis of sequences that are similar along their entire length is relatively simple. However, alignment of most sequences requires the positioning of gaps in the sequences involved. The fact that a gap of any length can occur within a sequence introduces the problem of judging how many individual changes have occurred within the organism and in what order. These gaps can thus be used as phylogenetic markers. Knowledge of changes within an organism through mutation over time give us knowledge of regions within a sequence that have not changed. These conserved regions, that may be present in a particular protein might therefore give away the function of this protein. Focusing on the conserved regions of sequences obtained from different levels in the phylogenetic tree, one might thus be able to predict the structure through homology modeling with another known protein with a similar function. Structural similarity will then aid in the rational design of ligands as drug compounds which might prove to have a high binding affinity with these structurally conserved regions, thus helping produce more efficacious and specific drug compounds.

7.1.5 Phylogenetic Tree Reconstruction

Phylogenetic analyses have their own set of problems though. One of the main ones is that of phylogenetic tree reconstruction. The main purpose of phylogenetic tree reconstruction is to predict or make an estimate on the phylogeny for some input data. For any collection of data, there will be some ancestral relationship among the sampled sequences. The data itself contains information that can

be used to reconstruct or to infer these ancestral relationships. This involves the reconstruction of the phylogenetic tree which illustrates the relationship among the sequences. There are two types of input methods that can be used to reconstruct the phylogenetic tree, one is character based(Maximum Parsimony and maximum likelihood) and the other is distance based(Unweighted Pair Group with Arithmetic Mean- UPGMA, Transformed Distance Methods and Neighbour Relation Methods). These methods will be explained further later.

7.1.6 Rooted and Unrooted Trees

Most methods for the inference of phylogeny yield trees that are unrooted. Thus from a tree itself, it is impossible to tell which species branched off before the others. Examples of a rooted and an unrooted tree are illustrated in Figure 7.2. To root a tree one should add an outgroup to the data set. An outgroup is a species for which external information (eg. paleontological information) is available that indicates that the outgroup branched off before all other species. Using more than one outgroup(they must not be closely related), we can improve the estimate of the final tree topology. However, in the absence of a good outgroup the root may be positioned by assuming approximately equal evolutionary rates over all the branches. In this way the root is put at the midpoint of the longest pathway between two species. This way of rooting is called mid-point rooting.

7.2 Parsimony

7.2.1 What is Parsimony

The principle of parsimony is defined as "a scientific rule which states that if there exists two answers to a problem or a question, and if, for one answer to be true, well-established laws of logic and science must be re-written, ignored, or suspended in order to allow it to be true, and for the other answer to be true no such accommodation need to be made, then the simpler of the two answers is much more likely to be correct." Putting it in a simpler way, parsimony is *a principle that states that the simplest explanation that explains the greatest number of observations is preferred to more complex explanations.*

Parsimony is one of the most popular character based method for the phylogeny reconstruction problem. The idea behind parsimony is to build the phylogeny with the fewest point mutations.

Formal Definition:

- Let S be a set of (DNA or Protein) sequences.

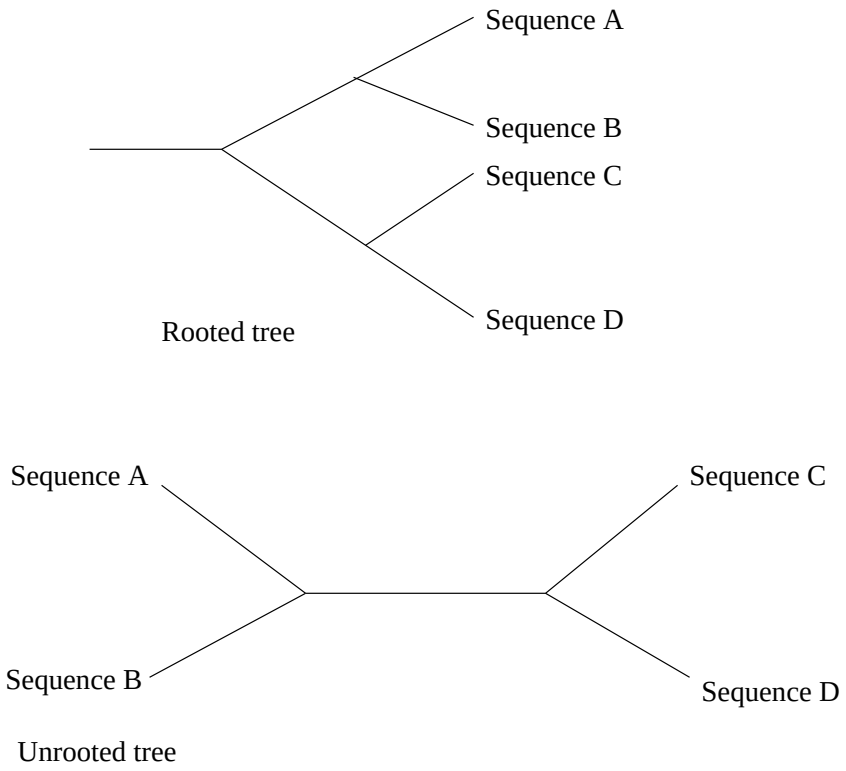


Figure 7.2: Structure of evolutionary

- Denote H(x,y) be the hamming distance between two sequences x and y.
- The most parsimonious tree is a tree T leaf-labeled by S and each internal node is assigned a sequence such that $H(T) = \Sigma_{(x,y)\in E(T)} H(x,y)$ is minimized. Note that H(T) is called the parsimony length of T.

Figure 7.3 shows an example of the most parsimonious tree for four species where each is represented by a sequence of 4 characters.
Note: Although there is no rule that requires nature to follow the simplest path, and results can vary based on which pieces of evidence are used, Parsimony is nonetheless a strong basis for the scientific work of paleontologists.

7.2.2 Computational Problems

The problems related to the parsimony approach can be categorized in two groups:

- Small Parsimony problem: To find the parsimony length of a given tree topology.
- Large Parsimony problem: To find the most parsimonious tree.

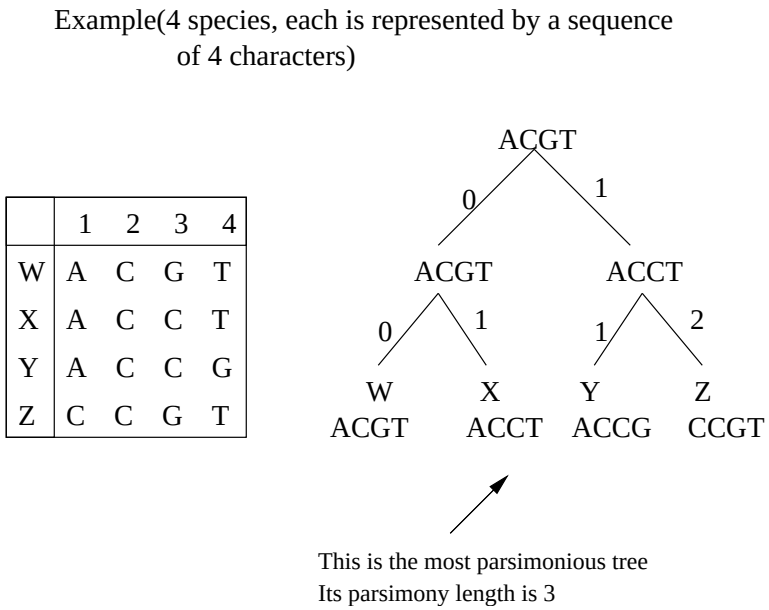


Figure 7.3: Example of parsimonious tree

7.2.2.1 Small Parsimony Problem

Input: Given a set S of sequences and the topology of a rooted phylogenetic tree T with leave labeled by S
Goal: Find parsimony length of T.
Problem statement:

1. What is the minimum number of changes for this topology?
2. What is the optimal labeling of the internal nodes?

This problem can be solved using Fitch's algorithm [F71] in polynomial time. First of all, it is clear that the problem can be solved for each character separately. As characters are mutually independent, the idea can be extended to handle sequences of n characters.

Each sequence has one character(Simple case)
Input: a leaf-labeled tree T where each leaf v is labeled by a single character v_c.
Output: a fully-labeled tree which is also the most parsimonious tree of T.

Algorithm:
The algorithm first puts all characters at the leaves of the tree and then it tries to label every internal node of tree, such that every node has label equal to one of its children. Formally the algorithm is as follows.

1. For every leaf v, let $S_v = \{v_c\}$.
2. For every internal node v with children u, w,
   let $S_v = S_u \cap S_w$ if $S_u \cap S_w \neq \Phi$; and
   $S_u \cup S_w$ otherwise
3. For every node v in preorder,
   - Let u be its parent. If $u_c \in S_v$ set $v_c \leftarrow u_c$; otherwise, assign any character in S_v to v_c.

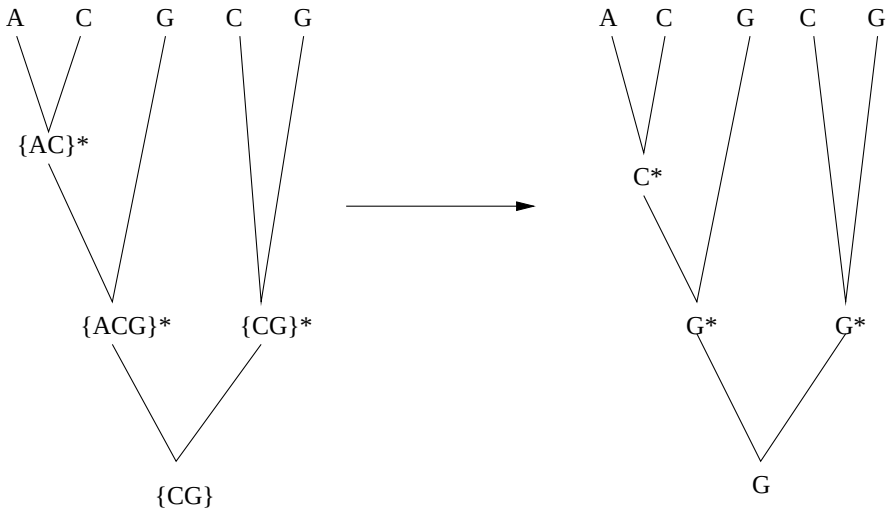


Figure 7.4: Small parsimony problem(simple case)

Figure 7.4 shows an example for the implementation of the above algorithm. Time complexity of the above algorithm is O(nk) where k is the size of the sequence.

Each sequence has m characters: For solving the problem with each sequence having m characters, it should be noted that the i^{th} character and the j^{th} character are independent for any i and j. Thus, the problem of having m characters in a sequence can be solved using m instance of the simple case problem. Hence the time complexity would be O(mnk).

Even after having solved the problem of small parsimony, there is still a long way to go, because the final goal is to find the optimal phylogeny.

7.2.2.2 Large Parsimony Problem

Input: A set S of sequences.
Output: The most parsimonious tree.
Problem statement:

1. What is the optimal phylogeny for these species, i.e., the one minimizing the parsimony score?

The large parsimony problem is a NP-hard problem. It can be 2-approximated in polynomial time, using the following algorithm.

Approximation algorithm:

- Given a set S of sequences, define G(S) be a weighted complete graph whose nodes are labeled by S and each edge (i, j) has weight H(i, j), where H(i, j) is the hamming distance between sequences i and j.
- return the minimum spanning tree of G(S).

Theorem 7.1 *Let T be a minimum spanning tree of G(S). Then, the parsimony length of T is at most twice that of the most parsimonious tree.*

Proof:

- Let T* be the most parsimonious tree.
- Let C be an Euler cycle of T*.
- Let P contains only the nodes of G(S) ordered in the way in which they appear in C.
- $w(T) \leq w(P) \leq w(C) = 2$ w(T*)

■

It should be noted that the maximum parsimony is statistically inconsistent i.e given a long enough sequences, maximum parsimony may not be able to recover the true tree with arbitrarily high probability. In other words, it may not be able to converge to the same tree.

7.3 Compatibility

Compatibility is another attempt to define a target function for the phylogeny problem:Given a set of characters which represents a set of species, it tries to find a phylogeny tree which is compatible with as many characters as possible. In fact, it is a simplification of parsimony approach. In the following discussion, we assume each character in the sequence can only have binary states, say 0 and 1 without loss of generality.

7.3.1 Problem definition

Let's have some basic definitions, before defining the compatibility problems:

7.3.1.1 Compatible

A binary character(a sequence of binary bits) c is compatible to a leaf-labeled tree T if and only if there exist an assignment of states to the internal nodes of T such that a change of state exists at most in one edge.

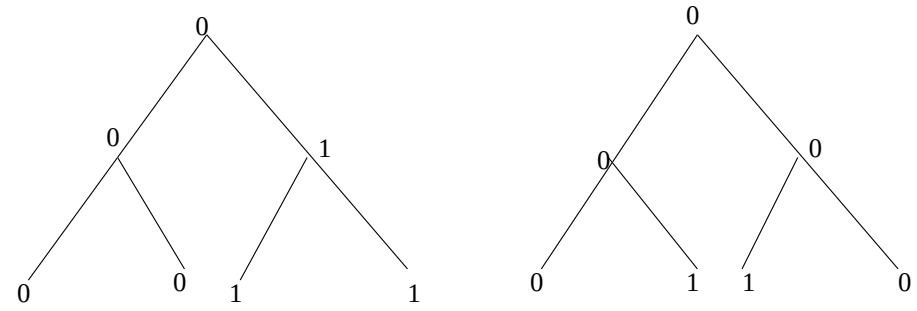


Figure 7.5: compatibility of a character with a tree

Figure 7.5 shows an example of compatibility and non-compatibility. When only one change is found in the edges of the tree, the character is compatible, otherwise, if there are more than one change in the edges of the tree, the character is incompatible to the tree. In fact, there is another way to express the compatible property. If a character c is compatible to a tree T, then an edge(u,v) can be identified in T, such that

- All the leaves in the subtree rooted by v have state s in the corresponding positions in the character c.

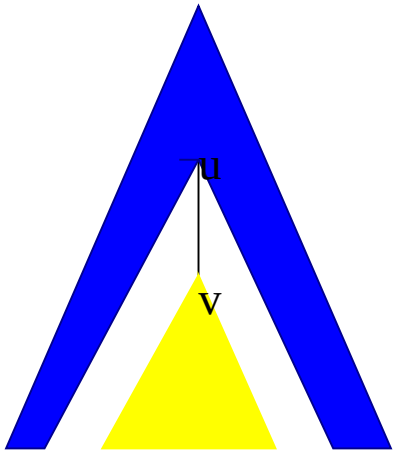


Figure 7.6: Compatible example for compatible case

- The other leaves have state (1-s) for the remaining positions in the character c

This property can be seen from Figure 7.6. The reason is quite straightforward, since only one edge can have change, say this edge is (u,v), without lose generality, say v is a son of u. Then the leaves in the subtree rooted by v must have the same state, otherwise, other changes will exist. The same reason for the rest of leaves.

7.3.1.2 Perfect Phylogeny

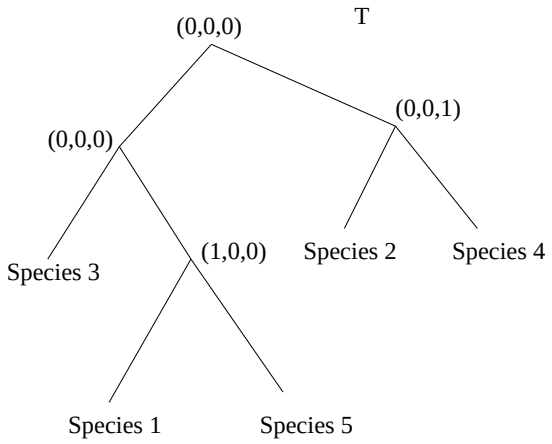
Based on compatibility concept, perfect phylogeny can be defined as follows: **Input:** Given n species, each is characterized by m binary characters. The input can be represented using a binary matrix M with n rows and m columns,where M_{ij} is the state of character j of species i. **Definition:** M admits a perfect phylogeny if and only if there exists a rooted tree T for n species such that all m characters are compatible.
Figure 7.7, shows an example for the input matrix M. As shown in the figure,the matrix M admits a perfect phylogeny since X1,X2 and X3 are all compatible to the tree T.

7.3.1.3 Computation Problems based on compatibility

There are two kinds of phylogenetic problem based on compatibility:

- **Compatibility problem** Given n species, each characterized by m binary characters,represented by a binary matrix M, compatibility problem is to check whether this set of species admits a perfect phylogeny.
  The problem is to answer whether there exists a rooted tree T such that all m characters are compatible to the tree T.

M	X1	X2	X3
Species 1	1	1	0
Species 2	0	0	1
Species 3	0	0	0
Species 4	0	0	1
Species 5	1	0	0



Example: characters 1, 2 and 3 are all compatible!

Figure 7.7: Compatiblitiy example for compatible case

- **Large Compatibility Problem(Perfect Phylogeny Problem)** The input for large compatibility problem is the same as compatibility problem, we are given n species, each characterized by m binary characters, which is represented using a binary matrix M. Large compatibility problem is to find a maximum set of characters which admits a perfect phylogeny.
  So, in the case if M can't admit a perfect phylogeny, its maximum subset could be found, which can admit a perfect phylogeny.

7.3.2 Compatibility Problem

Compatibility problem can be divided into two steps.

- In the first step, check whether M admits a perfect phylogeny.
- In the second step, if M admits a perfect phylogeny, recover the tree.

These two steps are straightforward, the first step only check whether a perfect phylogeny exists, the second step then finds the tree in case of existence.

7.3.2.1 Majority Rule

Observe that if M admits a perfect phylogeny T, after exchange 0 and 1 in any column, the result matrix M still admits the same perfect phylogeny T.
Figure 7.8 shows an example of such exchanging on column X_2.
Based on the above observation, we can define the **majority rule** as:
If there exists a column such that the number of state 1 > number of state 0, exchange 0 and 1.
As shown in observation, this transformation has no effect on compatibility.

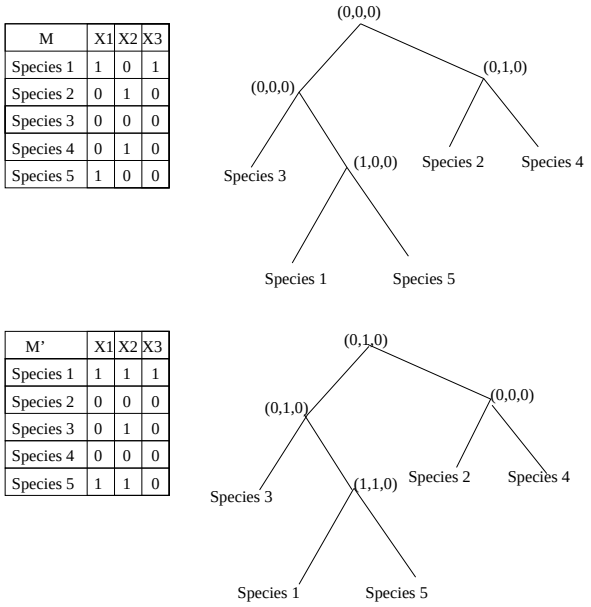


Figure 7.8: Exchanging states in a column

7.3.2.2 Main lemma in Compatibility

There are some properties of the compatibility. They will be useful in both simple solution and the advance solution afterward.Before describing the lemma, let's make some definitions.
For every character i, let O_i be the sets of species with state 1.
For example, in Figure 7.7, O_1=$\{1,5\}$
Characters i and j are *pairwise compatible* if they satisfy:

- O_i and O_j are disjoint or one of them contains the other.

For example, in Figure 7.7,O_1=$\{1,5\}$ O_2=$\{1\}$ So,O_1 contains with O_2 ;
Furthermore,O_3=$\{2,4\}$ so,O_1 is disjoint with O_3.

Lemma 7.2 *Let M be a binary matrix after applying the "majority rule". M admits a perfect phylogeny if and only if for every characters i and j, they are pairwise compatible.*

Proof: *M admits a perfect phylogeny* $\Rightarrow$ *every characters i and j are pairwise compatible.*

- Given that M admits a perfect phylogeny.

- Since we have applied "majority rule" to M, $|O_i| \leq n/2$ for every character i.
- Assume that i and j are not pairwise compatible.
- That is, there exists three species X,Y,Z such that Y,Z$\in O_i, X \notin O_i$ and $X, Z \in O_j, Y \notin O_j$.
- Since $O_i \cap O_j$ is non-empty, thus, $|O_i \cup O_j| < n$.Thus, there exists a species W$\notin O_i, O_j$.
- By character i, WLOG, Y and Z are in the same partition in T, while X and W are in another partition.
- By character j, X and Z are in the same partition in T , while W and Y are the same partition in T.
- Above situation can never happen in a T.
- Impossible, we arrived at Contradiction!

Every characters i and j are pairwise compatible ⇒ M admits a perfect phylogeny.
Prove: as Exercise. ■

7.3.2.3 Simple Solution for Checking Whether M Admits a Perfect Phylogeny

Based on the previous lemma, we can get the following algorithm easily. The algorithm is described as following:

- Apply "Majority Rule" to M.
- For every characters i and j,
  - Check whether i and j are pairwise compatible
  - If no, return "can't admit a perfect phylogeny!"
- Return "Admit a perfect phylogeny!"

From the lemma, it is known that, if every characters i and j are pairwise compatible, the matrix must admit a perfect phylogeny. So we just check every characters i and j, construct O_i, O_j, to see whether they are disjoint or contained.
The construction of these set can be done in O(n) time for each i and j, the checking also need O(n) time. Totally, we need check $\frac{m^2}{2}$ pairs. So totally, the time complexity is $O(m^2 n)$.

M	X1	X2	X3
Species 1	1	0	1
Species 2	0	1	0
Species 3	0	0	0
Species 4	0	1	0
Species 5	1	0	0

$|O_1|=2$,
$|O_2|=2$,
$|O_3|=1$

Figure 7.9: Step 1

7.3.2.4 Advance Solution for Checking Whether M Admits a Perfect Phylogeny

We just addressed a simple solution for compatibility problem with time complexity $O(m^2n)$. An intuitive question is , can we do better? The answer is yes, an O(mn) time algorithm can be obtained.
The idea is as follows:

- Show that there exists an algorithm in O(mn) time which can check all i,j, whether O_i and O_j are disjoint or one of them contains the other.
- If the condition is satisfied, M admits a perfect phylogeny;Otherwise, M does not admit a perfect phylogeny. The total time is O(nm) in this approach.

Now let's consider the following example.

Assume we are given 5 Species with 3 set of values which are binary in nature.
In Step 1: Relabel the characters so that $|O_i| \geq |O_j| if i < j$.
We can see that the sequence is already in such sorted order, thus we can proceed to step 2 without any changes.
In Step 2:

for every species i and character j, **do**
If $M_{ij} = 1$, let L_{ij} be the biggest $k < j$ such that $M_{ik} = 1$. If such k does not exist, $L_{ij} = -1$
If $M_{ij} = 0$, let $L_{ij} = 0$.
end for

Above figure illustrate how we get L from M
For example M_{11} has value 1, but there is no smaller value of k that $M_{ik} = 1$, thus it $L_{ij} = -1$. Another example L_{23} has value 1 because $M_{21} = M_{23} = 1$

M	X1	X2	X3
Species 1	1	0	1
Species 2	0	1	0
Species 3	0	0	0
Species 4	0	1	0
Species 5	1	0	0

L	X1	X2	X3
Species 1	-1	0	1
Species 2	0	-1	0
Species 3	0	0	0
Species 4	0	-1	0
Species 5	-1	0	0

Figure 7.10: Step 2

For the rest of those M_{ij} whose value is 0, according to the algorithm, it remain as 0 in the L matrix.

Below **Technical Lemma**, tells us the use of the matrix L.

Lemma 7.3 *For some character j, if there exist two nonzero entries L_{ij} and L_{kj} such that $L_{ij} \neq L_{kj}$, then M does not admit a perfect phylogeny.*

Proof:

- Suppose $L_{ij} = x$ and $L_{kj} = x'$. WLOG, $x > x'$.
- By definition, $M_{ij} = M_{kj} = 1, M_{ix} = 1, M_{kx} = 0$
- Thus, O_j contains species i and species k and O_x contains species i, but not species k. It means that (1) $O_j \cap O_x \neq \phi$, (2) O_j is not subset of O_x
- Note that $j > x$. Thus, $|O_x| \geq |O_j|$
- As $k \notin O_x, O_x$ should contain some species which does not appear in O_j. So, (3) O_j is not subset of O_x.

By Lemma 7.2, M does not admit a perfect phylogeny.

■ **In Step 3:**

for every character j, check if there exist i and k such that $L_{ij} \neq L_{kj}$ and both L_{ij} and L_{kj} are nonzero **do**

If yes, return "does not admit a perfect phylogeny".

end for

Return "admits a perfect phylogeny".

From the matrix L, in a glance we can see that it doesn't fulfill rule (i), for all nonzero in each of the character X_1, X_2 and X_3. All the nonzero for X_1 and X_2 are -1 and there is no other nonzero value, and all nonzero in X1 are 1.

Thus for every character j (column j), we can't find two nonzero entries which are different. So, for all i, j, O_i and O_j are disjoint or one of them contains the

L	X1	X2	X3
Species 1	-1	0	1
Species 2	0	-1	0
Species 3	0	0	0
Species 4	0	-1	0
Species 5	-1	0	0

Figure 7.11: Step 3

other. So this means that the matrix is a perfect phylogeny.

Time complexity:

In terms of time complexity,

1. Step 1 takes O(mn) time using radix sort.
2. Step 2 and 3 can be computed in O(mn) time.

Radix Sort is a sorting algorithm that takes linear time, thus step 1 takes O(mn) time. Step 2 and 3 can be computed in O(mn) time, as just need to scan through every column j and every row i once. Thus, we can decides whether M admits a perfect phylogeny or not in O(mn) time.

7.3.3 Reconstruct Perfect Phylogeny from M

After we have detected whether M admits a perfect phylogeny tree, we can now construct the perfect phylogeny using below algorithm.

Algorithm

- Let T be a tree containing a single root node r. N(r)={1,...,n}
- For every character j where j=1 to m
  - Find a node $v \in T$ such that

M	X1	X2	X3
Species 1	1	0	1
Species 2	0	1	0
Species 3	0	0	0
Species 4	0	1	0
Species 5	1	0	0

Figure 7.12: Matrix M

  - ∗ N(v) can be partitioned into two non-empty sets N_0 and N_1 where N_s ={ x ∈ N(v)| character j of species x is of state s } for s=0,1 /* Note: we can only split one node v */
  - Create two children v_0 and v_1 for v
  - Set $N(v_0) = N_0, N(v_1) = N_1$
  - Set $N(v) = \phi$

- For every node v s.t. N(v) is nonempty,
  - If $|N(v)| > 1$, let the species in N(v) be the children of v
  - If $|N(v)| = 1$, node v represents the species in N(v)

As an example, consider the 5 species with 3 character (X_1, X_2 and X_3), which according to the previous rule, it admits a perfect phylogeny tree.

- In the initial case, we list down all species in a node.
- Using Character 1, X_1 as a base, we split the species into 2 partitions, the left side node holds all species whose character has state 1 while the right side node holds all species whose character has state 0.
- The step is follow by Character 2 splitting, but there is no change to the tree at this stage as in the example, character 1 and 2 are complement of each other.
- In the next stage character 3 is split, dividing 1 and 2,4.
- To complete the phylogeny tree, each species has to be a leaf and not a node itself, so the node is again split and for a new leaf for the species.

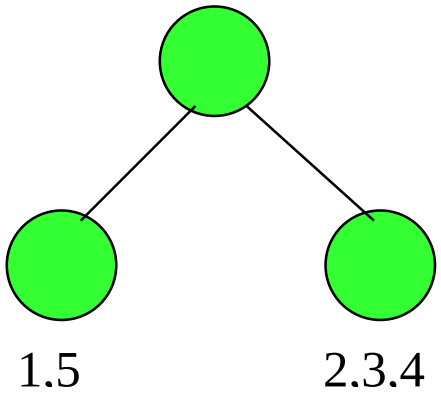


Figure 7.13: character 1

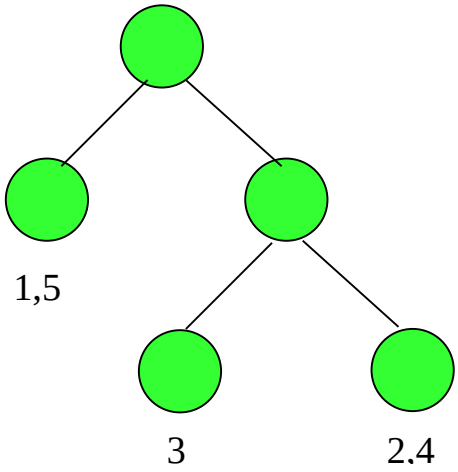


Figure 7.14: character 2

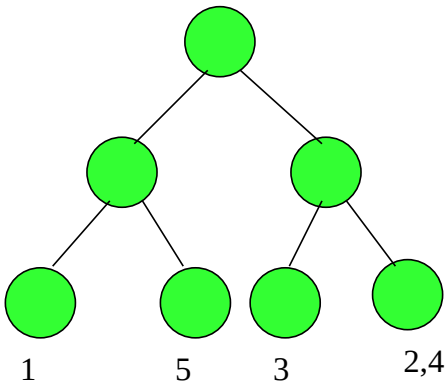


Figure 7.15: character 3

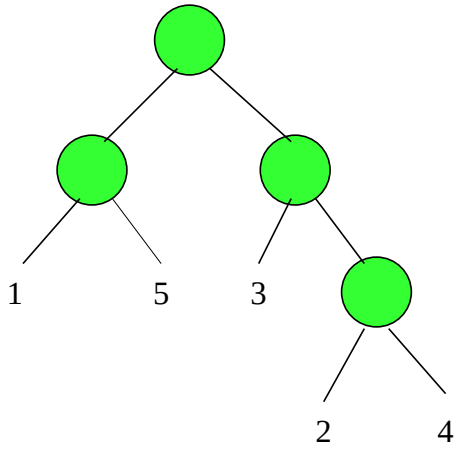


Figure 7.16: final

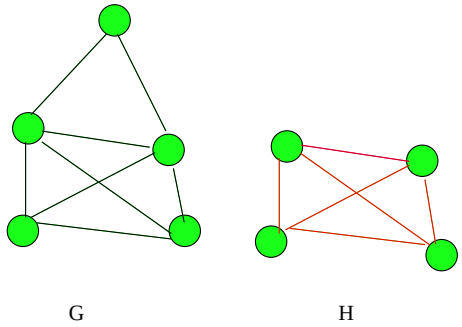


Figure 7.17: CLIQUE problem

In the time analysis, For every character j, it takes O(n) time to identify a node and to split the node. Thus, the total time taken is O(nm).

7.3.4 Large Compatibility Problem

The difficulty in solving the large compatibility problem is to find the maximum set of species which admits a perfect phylogeny, which is NP-hard.

One way to solve the large compatibility problem is by transforming it to the CLIQUE Problem, which is defined as follows.

Given a graph G, the CLIQUE problem tries to find the maximum size subgraph H such that H is a complete graph. For example, in figure below, the maximum size complete subgroup of G is in fact H.

7.3.4.1 Algorithm for solving large compatibility problem

Input: the instance M;
1: Applying the "majority rule" to M;
2: Obtain G based on M;
3: Find the maximum clique in G;
4: Recover the maximum subset of compatible characters;
5: Based on the tree construction algorithm described in last section, recover the phylogeny

From the description of the algorithm, we may discover that the bottleneck is step2. Since the CLIQUE problem is NP-Complete, the time complexity of the algorithm is still exponential.

7.3.5 Compatibility for characters with k possible states

Now we generalize the problem when the characters are not binary as following:

Definition 7.4 *A character c with k possible states is compatible to a leaf-labeled tree T, if and only if there exist an assignment of states to the internal nodes of T such that the total number of state changes is exactly $k-1$.*

The two corresponding problems for multiple states cases as following.

- Compatibility Problem
  - When the number of states is constant, polynomial time algorithm is still feasible;
  - When the number of states is variable, NP-Complete.
- Large Compatibility Problem: NP-Complete

7.4 Maximum Likelihood Based Model

In this section we study a maximum likelihood based model of phylogenetic trees reconstructions.

7.4.1 What is a Likelihood Based Model

Give a set of data D, maximum likelihood tries to find a model M such that $Pr(D|M)$ is maximized. A model consists of two components. They are

- A rooted tree which models the evolution relationship;
- Every edge is associated with a stochastic model of evolution.

And we have two assumption for the stochastic model of evolution. They are

- The characters evolve identically and independently;
- The trees has the Markov property. That is, the evolution occurs at one subtree is independent to the other parts of the trees.

And examples of models include Cavender-Felsenstein model (also called Cavender-Farris model)[CF87], and Jukes-Cantor model [JC69]. In this lecture, we study the former one in detail.

7.4.2 Cavender-Felsenstein Model

We now introduce the **Canveder-Felsenstein Model** [CF87] in details. The Cavender-Felsenstein Model is the *simplest possible Markov model* of evolution.

7.4.2.1 Definition of the Model

The model assumes that each character has two states. Cavender-Felsenstein Model consists of the following.

- The topology of a rooted evolution tree T;
- A stochastic model of evolution for every edge e, that is, for every character x_i, a mutation probability $p_i(e)$ for each edge e in T.

For the stochastic model of evolution, we have following assumption.

- $\forall x_i, \forall e = (u, v) \in T$, we have $0 < p_i(e) < 0.5$, and

	$u = 0$	$u = 1$
$v = 0$	$Pr(u = 0\|v = 0) = 1 - p_i(e)$	$Pr(u = 1\|v = 0) = p_i(e)$
$v = 1$	$Pr(u = 0\|v = 1) = p_i(e)$	$Pr(u = 1\|v = 1) = 1 - p_i(e)$

- $Pr(u|v) = Pr(v|u)$;
- For the root r, $Pr(r = 0) = Pr(r = 1) = 0.5$.

Thus given a data set D_i, we may use (T, p_i) to represent a model, where T is the topology of the evolution tree, and $p_i : E(T) \to (0, 0.5)$ is the mutation probability for each edge in T, where $E(T)$ is the edge set of T.

We denote $Pr(D_i|T, p_i)$ the probability that the data is D_i given that the model is (T, p_i). Now we present an example for calculating $Pr(D_i|T, p_i)$.

Consider three species a, b, and c. Assume the tree topology T of the model is as shown in Figure 7.18.

For a particular character i, assume the mutation probability for every edge e is $p_i(e)$;

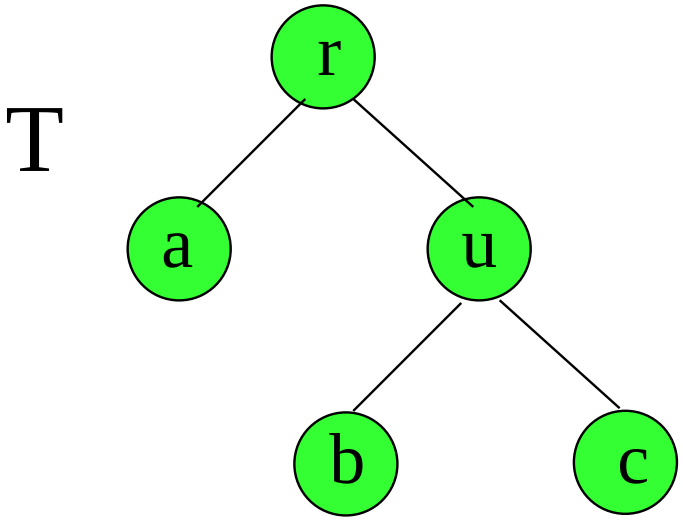


Figure 7.18: Cavender-Felsentein Model

Suppose the data D_i says: $a_i = 1$, $b_i = 1$, $c_i = 0$;

Then, we obtain the probability that the data is D_i given that the model is (T, p_i), $Pr(D_i|T, p_i)$, equals

$$\sum_{k=0,1;j=0,1} Pr(r_i = k)Pr(a_i = 1|r_i = k)Pr(u_i = j|r_i = k)Pr(b_i = 1|u_j = 1)Pr(c_i = 0|u_i = j).$$

Now we may obtain the probability that the data is $D = D_1 \cup ... \cup D_n$, give that the model is $(T, p_1, ..., p_n)$ as follows.

Theorem 7.5 *Given the data $D = \cup_{1 \le i \le n} D_i$ and the model $(T, \{p_i | 1 \le i \le n\})$, where T is the topology of the evolution and p_i is the mutation probability for character i, we obtain that*

$$Pr(D|T, p_1, .., p_n) = \prod_{i=1}^{n} Pr(D_i|T, p_i). \tag{7.1}$$

7.4.2.2 Computational Problems

There are two computational problems for Cavender-Felsenstein model.

- Likelihood of a model
  Give the model M, for any data D, try to compute $Pr(D|M)$;

- Find model with maximum likelihood
  Give data D, try to find a model M which maximizes $Pr(D|M)$.

We now discuss the algorithms of computing these two problems as following.

7.4.2.3 Likelihood of a model

We may describe the the computation problem of the likelihood of a model as

- Input

  Data D: m species where each species is characterized by n character;

  Model M: $M = (T, p_1, ..., p_n)$;

- Aim

  Compute $Pr(D|M)$.

It is obvious that we can compute $Pr(D|M)$ using (7.1), but it will take exponential time. However, we can define the likelihood recursively and compute the value using *dynamic programming* as following.

- **Definition:** For a particular character i, let $L_i(v, x)$ be the likelihood of the subtree rooted at v, given that character i has state s.

- **Basis:** for allleaf v, and state s,

$$L(v, s) = \begin{cases} 1 & \text{if } v_i = s \\ 0 & \text{otherwise.} \end{cases} \tag{7.2}$$

- **Recurrence:** Traverse the tree in postorder, for every internal node v with children, says, u and w,

$$L_i(v, s) = \left[\sum_{y=0,1} L_i(u, y) Pr(u_i = y|v_i = s)\right] \left[\sum_{y=0,1} L_i(w, y) Pr(w_i = y|v_i = s)\right] \tag{7.3}$$

- **For the Root:** Finally, for the root, we have

$$L = \prod_{i=1}^{m} \left[\sum_{s=1,2} \left(\frac{1}{2} L_i(root, s)\right)\right] \tag{7.4}$$

Now we analyze the time complexity of the dynamic programming algorithm.

- For every node v and every state s, $L_i(v, s)$ can be computed in $O(1)$ time according to (7.3);

- Since there are n nodes and m characters, all $L_i(v, s)$ can be computed in $O(mn)$ time;
- For L, it can be computed in $O(m)$ time;
- In total, Likelihood of a tree can be computed in $O(mn)$ time.

7.4.2.4 Find model with maximum likelihood

We may describe the the computation problem of the likelihood of a model as follows

- Input

  Data D: m species where each species is characterized by n character;

- Aim

  Find a model $M = (T, p_1, ..., p_n)$ which maximizes $Pr(D|M)$

This is a difficult problem, although we still don't know whether it is NP-hard or not. A practical solution is to use heuristic method to get close to optima (like [PAUP]).

7.4.3 Final remark for Maximum Likelihood

For the Cavender-Felsenstein model, maximum likelihood is statically consistent, i.e. given long enough sequence, for the Cavender-Felsenstein model, maximum likelihood is able to recover the true tree with arbitrarily high probability.

References

[F71] W. M. Fitch, "Toward defining the course of evolution: minimum change for a specified tree topology",*Systematic Zoology*, 20:406-416, 1971.

[CF87] J.A. Cavender, and J. Felsenstein, "Invariants of phylogenies: Simple case with discrete states", *Journal of Classification*, 4, 57-71. 1987.

[JC69] T. H. Jukes, and C. Cantor, "Mammalian Protein Metabolism", *chapter Evolution of protein molecules*, pages 21-132, Academic Press, New York, 1969.

[JW99] K. Junhyong Kim, and T. Warnow, "Tutorial on Phylogenetic Tree Estimation Tutorial for ISMB 99", 1999.

[PAUP] http://newfish.mbl.edu/Course/Software/PAUP/, Marine Biological Laboratory