Outline

Today we have two student lectures.

- Evolutionary Trees (*Andris Ambainis*)
  1. Parsimony
  2. Maximum-Likelihood
  3. Distance Methods
- Finding Authoritative Web Pages Using the Web's Link Structure (*David Wagner*)

1 Evolutionary Trees (Andris Ambainis)

An evolutionary tree, or *phylogeny*, is a tree which describes the ancestral relationships among a set of species. The leaves of an evolutionary tree correspond to the species being compared, and internal nodes correspond to (possibly extinct) ancestral species. Figure 143 shows a simple evolutionary tree for cats, dogs and lynxes. Domestic cats and lynxes share a feline ancestor distinct from dogs, but all three species share a common ancestor as well.

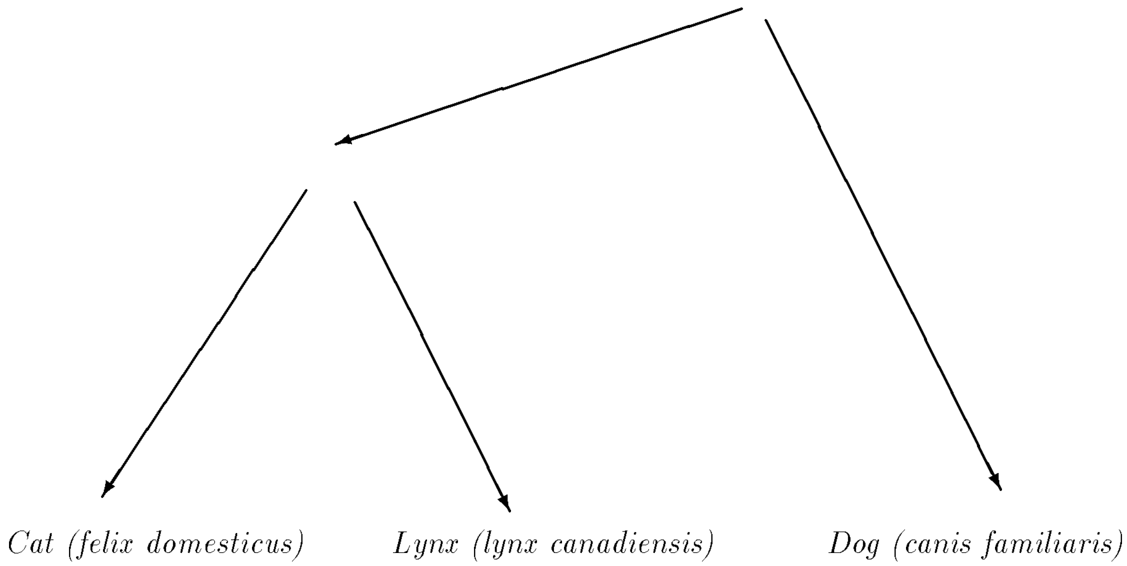

Figure 143: An evolutionary tree for cats, dogs and lynxes

Scientists have been studying and creating phylogenies for over a century. Traditionally, morphological information and taxonomies were used as data for creating evolutionary trees. This process was often guided more by intuition than a rigorous set of rules. Today, more systematic methods based on DNA sequencing are used. DNA sequencing provides a much larger body data which can be precisely analyzed. Such data both demands and facilitates computer algorithms for generating evolutionary trees.

Three algorithms for creating evolutionary trees will be discussed in the following subsections. In each case, we assume that the DNA sequences have already been aligned using multiple alignment, and that each position in the sequence is independent.

1.1 Parsimony

In the parsimony method, the best tree is the one which explains the data using the smallest number of evolutionary changes:

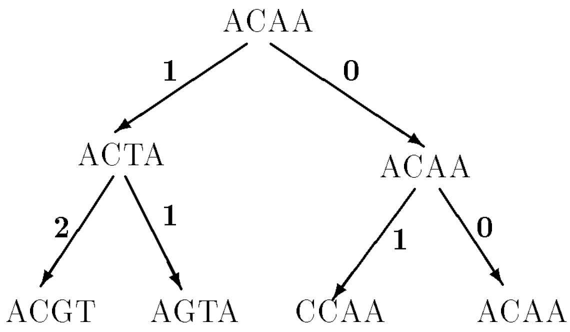

Figure 144: A parsimony model for sequences ACGT, AGTA, CCAA and ACAA. The numbers on the edges indicate the number of mutations between nodes

Parsimony can be formulated as a Steiner Tree problem. For example, given a set of m aligned DNA sequence of length n, consider the n dimensional hypercube $G = \{A, G, C, T\}^n$. The weights on the edges of this graph are the distance metric between the two endpoints. Each DNA sequence is a vertex on this graph. The problem of parsimony is to find the minimum steiner tree with these vertices as demand points.

Analysis

If we are given the tree structure with the leaves assigned, we can find a minimum-cost assignment for the internal nodes using dynamic programming. However, to find the tree that minimizes the number of changes is an *NP*-hard problem.

We can approximate the solution, however, using local search heuristics. First, we find a "good" tree. One way to do this is using a greedy method: starting with the set of species, keep connecting the two subtrees with the most closely-related roots until there is only one tree. We then apply rearrangement operations to the tree and check if the rearranged tree has a lower cost. Possible rearrangement operations include:

- Swap two subtrees that are children of neighboring nodes.

- Remove a subtree and attach it to some other node.

This process can be repeated a fixed number of times, or until we find a local minimum.

Discussion

Parsimony assumes that mutations are rare. In fact, this may not the case. Indeed, it is possible using parsimony to construct evolutionary trees that are "positively misleading" (*i.e.*, the probability of getting the wrong tree approaches one as the amount of data approaches infinity) [J78].

Despite being NP-hard and positively misleading, parsimony remains popular among evolutionary biologists. It has the advantage of being simple, intuitive and appealing. A recent study of Rice and Warnow shows that on artificial data, parsimony performs quite well [RW97].

1.2 Maximum-Likelihood

The maximum-likelihood method generates an evolutionary tree from a stochastic model of evolution. The same model can be used to generate trees for different sets of species.

To simplify this presentation, we first assume that DNA sequences are binary (*i.e.*, sequences of 0's and 1's).

We will describe stochastic models called *Cavender-Farris trees*. Such a tree has a probability associated with each edge as well as an initial state probability associated with the root. Given a tree with root probability p_R and probabilities p_e for each edge e, each position of the binary sequence is generated as follows:

1. The root is labeled 1 with probability p_R and 0 with probability $1 - p_r$.

2. Each labeled node broadcasts its label to all its children.

3. When a label is broadcast over edge e, it gets reversed with probability p_e.

Figure 145 shows an example of a Cavender-Farris tree. We generate a sequence for a node X (*i.e.*, species X) by repeatedly running the stochastic process and adding the label of X to the sequence, as illustrated in figure 146. This method relies on the assumption that all positions in a DNA sequence are independent.

When using four states (*i.e.*, A,G,C,T), each edge is assigned a probability matrix instead of a scalar probability p_e:

$$\begin{bmatrix} p_{\mathrm{AA}} & p_{\mathrm{AC}} & p_{\mathrm{AG}} & p_{\mathrm{AT}} \\ p_{\mathrm{CA}} & p_{\mathrm{CC}} & p_{\mathrm{CG}} & p_{\mathrm{CT}} \\ p_{\mathrm{GA}} & p_{\mathrm{GC}} & p_{\mathrm{GG}} & p_{\mathrm{GT}} \\ p_{\mathrm{TA}} & p_{\mathrm{TC}} & p_{\mathrm{TG}} & p_{\mathrm{TT}} \end{bmatrix}$$

where p_{XY} is the probability that X changes to Y. Biologists often assume that some or all p_{XY} are related, or even equal.

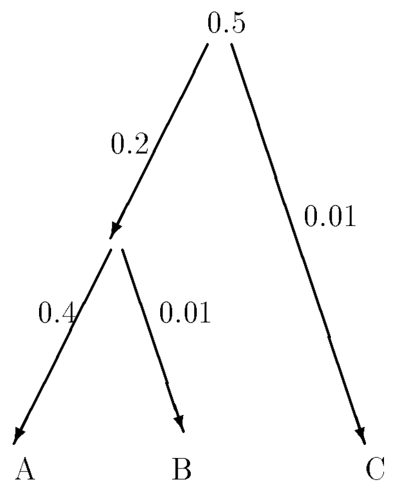

Figure 145: A stochastic model as a Cavender-Farris tree

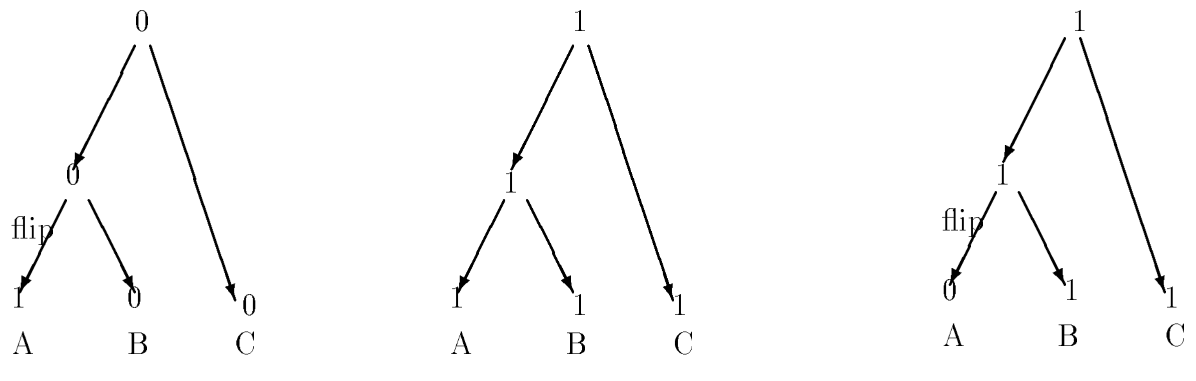

Figure 146: Generating sequences from the stochastic model in figure 145. Edges over which the label has been flipped are marked with "flip."

Given a stochastic model of evolution, the maximum-likelihood method finds a tree T that maximizes $p_T(S)$ where S is the set of DNA sequences and $p_T(S)$ is the probability that T will generate all the sequences in S.

This method is much more general than parsimony, which has a fixed model of evolution that may be incorrect. Assuming that the chosen stochastic model of evolution is accurate, this method will give a correct evolutionary tree for any set of species. However, it is the most computationally expensive of the methods discussed in this lecture, and is rarely used on large trees.

1.3 Distance methods

In the distance method, we create a matrix D consisting of similarity measures between pairs of species in the set being studied. Then, we find a tree that generates the distance matrix closest to D. This method is illustrated by the following example.

An example

Take three species, a, b and c with the following DNA sequences:

a: ACAAAAACAA
b: ACTTGAACAT
c: TCTAAAACAT

Now generate a similarity matrix for a, b and c:

$$\begin{bmatrix} - & 0.4 & 0.3 \\ 0.4 & - & 0.3 \\ 0.3 & 0.3 & - \end{bmatrix}$$

The similarity measure used is $d_{xy} = M/L$ where:
M is the number of positions with synonymous characters.
U is the number of positions with non-synonymous characters.
G is the number of positions with gaps in one sequence and residue in another.
W_G is the weight of gaps. (For this example $W_G = 1$)
$L = M + U + W_G G$

Transform the similarity matrix into a matrix with estimates of "evolutionary time" between species. We estimate the evolutionary time t_{xy} between two species x and y as $t_{xy} = \ln(1 - 2d_{xy})$.

$$\begin{bmatrix} - & 1.62 & 0.93 \\ 1.62 & - & 0.93 \\ 0.93 & 0.93 & - \end{bmatrix}$$

Now, find a tree with nodes a, b and c such that the distance between each of these nodes is the same as the evolutionary time between the corresponding species:
Note that the tree in figure 147 is undirected.

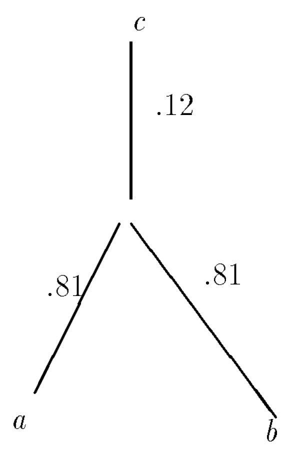

Figure 147: An evolutionary that fits the time matrix for species a, b and c.

Analysis

If the data precisely fits some tree, then this method is relatively simple. However, real data have random errors. If no tree exactly fits the data, finding the tree that is closest to a distance matrix is *NP*-complete under several definitions of "closeness." In practice, the following heuristic, called *nearest-neighbor joining* [SN87] is often used:

- Introduce closeness measure $d'(x, y) = (n-2)d(x, y) - \sum_x d(x, z) - \sum_y d(y, z)$
- Repeat until only one node remains:
  1. Search for x and y that minimize $d'(x, y)$.
  2. Replace x and y with a new node xy.
  3. Calculate $d(xy, z) = (d(x, z) + d(y, z) - d(x, y))/2$

The evolutionary tree will the tree where each pair of nodes x and y is connected its joint xy:

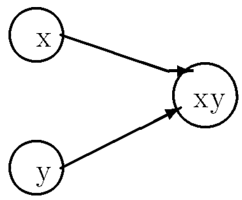

Figure 148: A branch of the evolutionary tree joining nearest neighbors x and y.

2 Finding authoritative web pages using link structure (David Wagner)

Today's web search engines do keyword searching. This technique is very powerful, but it handles broad queries quite poorly. The quantity of hits is often very high, and the quality of these hits varies widely.

We want to be able to identify authoritative sources for broad queries. This lecture presents an algorithm which:

1. Views the web as a directed graph.
2. Picks a relevant subgraph using standard search techniques.
3. Analyzes the resulting link structure.

Why analyze the link structure? Take two pages p and q such that p has a link to q. Through this link, the creator of page p has in some sense conferred authority on q. For example, the "Algorithms in the Real World" class page contains a large number of links to related pages. These linked pages can be considered authoritative sources on algorithms in the real world.

2.1 A naive approach using adjacency

The "authoritativeness" of p can be measured using the *in-degree* of p, or the number of pages which link to p. More precisely, we want the in-degree of p in a subgraph induced by the keyword search for the query. Such a subgraph can be generated by taking the top 200 pages returned by a standard search engine (like AltaVista or HotBot) as a root set, and then adding all "adjacent" pages (*i.e.*, pages with links into and out of the root set).

This approach does not work very well in practice. For example, a search on the word "java" will probably return pages on the programming language, the island in Indonesia and the popular caffeinated beverage. It is likely that the user only wanted one of these topics.

However, an analysis of the link structure in this query result will likely reveal three subgraphs, separated roughly by topic and with few interconnecting links between them. This property leads to an improved search algorithm.

2.2 An algorithm for finding communities within topics

Hubs are pages that point to multiple relevant authoritative pages. The best known example of a hub is Yahoo. An *authority* is a page that is pointed to by multiple relevant hubs. For example, the JavaSoft home page is an authority, since is linked by most hubs that catalogue Java links.

We want to find *communities*, or sets of hubs H and authorities A such that $H + A$ resembles a dense bipartite graph.

We will now define an iterative algorithm for finding communities. Page p is assigned weights $x[p]$ and $y[p]$ that represent authority weight and hub weight, respectively. The hub weight $x[p]$ is the sum of authority weights of its outgoing links. Likewise, $y[p]$ is the sum of the hub weights of pages that point to p. Figure 150 illustrates how $x[p]$ and $y[p]$ are calculated.

The algorithm can be implemented using linear algebra. Let A be the adjacency matrix of the graph induced by the keyword search. A_{pq} is 1 if p has a link to q and 0 otherwise. At

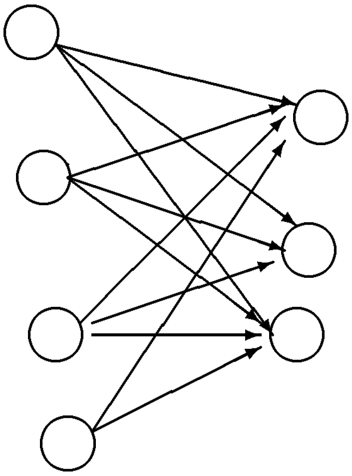

Figure 149: A subgraph representing a community. Hubs (on the left) point to authorities (on the right).

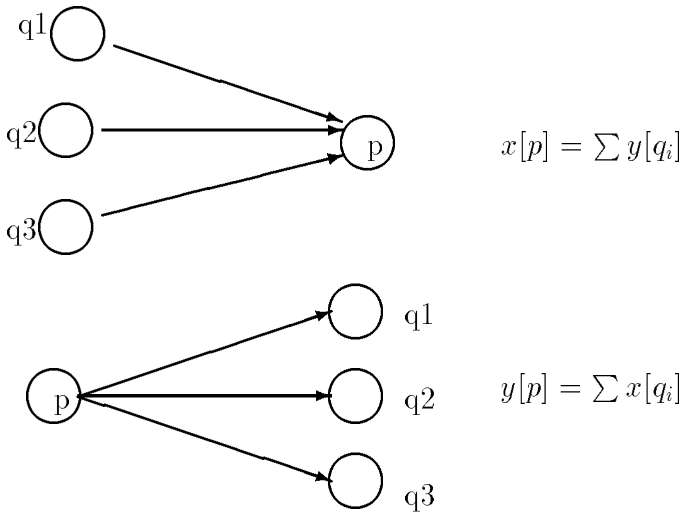

Figure 150: Calculating the hub weight and authority weight of a page p

each iteration, let $x = A^T y$ and $y = Ax$. Normalize x and y, and repeat the process until x and y converge. From linear algebra, we know that x and y will eventually converge to the principal eigenvectors of $A^T A$ and AA^T.

Each eigenvector corresponds to one community. The principal eigenvector corresponds to the most prominent community, but other eigenvectors can be used to examine other communities in the topic. It is interesting to note the similarities between the linear algebra approach to finding communities and the use of SVD in latent semantic indexing as described in the previous lecture. Recall that in finding the $SVD(A) = U\Sigma V^T$ that the columns of U are the normalized eigenvectors of AA^T and the columns of V are the normalized eigenvectors of $A^T A$. Using U_1 and V_1 gives us the principal eigenvectors (the ones with the highest eigenvalue). In fact, the iterative approach mentioned above (often called repeated squaring) is just one of the many approaches to solve for the first eigenvectors for the SVD.

Here are some examples of real queries using this algorithm:

- The principal eigenvector of the query "java" includes JavaSoft, the `comp.lang.java` FAQ, etc.
- The query "net censorship" includes the sites for EFF, CDT, VTW and ACLU, all of which are organizations that play a prominent role in free speech issues on the net.
- The query "abortion" returns pro-life pages as one eigenvector and pro-choice pages as another. Each side of the issue tends to form a bipartite graph with very few cross-links to the opposition.

This method works surprisingly well for broad queries, and shows promise for some other search tasks, such as "similar page" queries (*i.e.* search for pages with information similar to a given page). However, for very specific queries, the authoritative search does not work very well. Rather, it tends to find authorities for generalizations of the query topic. It might be possible to use lexical ranking function to keep the authoritative search oriented, but this is left open for future work.

For further reading

More details on the authoritative searching technique can be found elsewhere. [K97].

References

[J78] J. Felsenstein, "Cases in which parsimony or compatibility methods will be positively misleading." *Systematic Zoology*, 27:401-410, 1978.

[K97] J. Kleinberg. "Authoritative Sources in a Hyperlinked Environment." IBM Research Report RJ 10076, May 1997. This paper will also appear in the Proceedings of the ACM SIAM Symposium on Discrete Algorithms, 1998.

[RW97] Kenneth Rice and Tandy Warnow, "Parsimony is Hard to Beat." *COCOON97*.

[SN87] N. Saifou, M. Nei. "The neighbor-joining method: a new method for unconstructing phylogenic trees." *Mol.Biol.Evol.*, 4:406-425, 1987.