Phylogeny and Molecular Evolution

Character Based Phylogeny
Large Parsimony
Credit

- Ron Shamir’s lecture notes
- Notes by Nir Friedman Dan Geiger, Shlomo Moran, Sagi Snir
- Durbin et al.
- Jones and Pevzner’s presentation
- Bioinformatics Algorithms book by Phillip Compeau and Pavel Pevzner – all book photos shown in this lecture are from there.
Type of Tree Reconstruction

• **Character-based**
  • Input is a multiple alignment of sequences (one sequence per species).

• **Distance-based**
  • Input is a matrix of distances between species
  • Distance can be the relative length of the sequence which the two sequences disagree on, or alignment score between them, or (whatever)
Inferring a Phylogenetic Tree

**Generic problem: Optimal Phylogenetic Tree:**

- **Input:**
  - $n$ species, each represented by a length-$m$ string
  - set of characters,
  - for each species, the state of each of the characters.
  - (parameters)
- **Goal:** find a fully-labeled phylogenetic tree that best explains the data. (maximizes a target function).

**Assumptions:**
- characters - mutually independent
- species evolve independently

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>CAGGTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>CAGACA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>CGGGTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>TGCACT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>TGCCTGA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Parsimony Approach to Evolutionary Tree Reconstruction

- Applies Occam’s razor principle to identify the simplest explanation for the data
- Assumes observed character differences resulted from the fewest possible mutations
- Seeks the tree that yields lowest possible **parsimony score** - sum of cost of all mutations found in the tree
Small Versus Large Parsimony

- Small Parsimony – Topology (Tree and assignment of species to leaves) are given in advance. Polynomial Time algorithms (Fitch and Sankoff)
- Large Parsimony – The Topology is not known in advance. NP Hard – requires enumeration of all possible topologies.
Small Parsimony Problem

• **Input**: Tree $T$ with each leaf labeled by an $m$-character string.

• **Output**: Labeling of internal vertices of the tree $T$ minimizing the parsimony score.

• We can assume that every leaf is labeled by a single character, because the characters in a string are independent.
Extension to Many Letters

- What is the parsimony score of

A: CAGGTA
B: CAGACA
C: CGGGTA
D: TGCACT
E: TGCGTA

When the tree is known, we can do it character after character; each score is computed independently of the others.
Small Parsimony
Minimizing Total Hamming Distance

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Fitch Algorithm (Tree is Given)

• Work on each position in a string independently.
• Start at the leaves.
• If two children have common character, parent “inherits” it.
• Record union and go up.
• After reaching root, go down to fix sets of size > 1.
Weighted Parsimony

**Weighted Parsimony score:**
- Each change is weighted by a score $c(a,b)$.
- The weighted parsimony score reduces to the parsimony score when $c(a,a)=0$ and $c(a,b)=1$ for all $b \neq a$.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>T</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>G</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>9</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>
Weighted Parsimony on a Given Tree (Sankoff)

Each position is independent and computed by itself. Use Dynamic programming on a given tree.

- if \( k \) is a node with children \( i \) and \( j \), then
  
  \[
  S(k,a) = \min_b(S(i,b)+c(a,b)) + \min_d(S(j,d)+c(a,d))
  \]

\( S(j,d) \) = the score of subtree rooted at \( j \) when \( j \) has the character \( d \).
Evaluating Parsimony Scores

Dynamic programming on a given tree

**Initialization:**
- For each leaf \( i \) set \( S(i,a) = 0 \) if \( i \) is labeled by \( a \), otherwise \( S(i,a) = \infty \)

**Iteration:**
- if \( k \) is a node with children \( i \) and \( j \), then
  \[
  S(k,a) = \min_x (S(i,x)+c(a,x)) + \min_y (S(j,y)+c(a,y))
  \]

**Termination:**
- The **cost** of the tree is \( \min_x S(r,x) \) where \( r \) is the root

**Comment:**
- If we keep in each node for each character “a” the two characters \( x, y \) that bring about the minimum, then we can trace the best assignment to all internal nodes.
Large Parsimony

A character-based method

**Input:**

n sequences (one per species), all of length m.

**Goal:**

- Find a tree with the input sequences at its leaves, and an assignment of sequences to **internal nodes**, such that the total number of **changes** is minimized.
Example

**Input**: four nucleotide sequences: AAG, AAA, GGA, AGA taken from four species.

**Question**: Which evolutionary tree best explains these sequences?

**One Answer** (*the parsimony principle*): Find a tree that has a minimum total number of substitutions of symbols between species and their originator in the phylogenetic tree.

![Evolutionary tree diagram](image)

Total #substitutions = 4
Example Continued

There are many trees possible. For example:

The total number of changes is called the **parsimony score**.
Example Continued

There are many trees possible. For example:

The total number of changes is called the **parsimony score**.
Parsimony Based Reconstruction

Two separate components:

1. A procedure to find the minimum number of changes needed to explain the data (for a given tree topology)

2. A search through the space of trees.

3. We already say that (1) is easy, we will now show that (2) is hard.
Exploring the Space of Rooted Trees

taxa (n) # trees
4  15
5  105
6  945
8  135,135
10 30,405,375
Inferring a Phylogenetic Tree

Trees at level k-1 have $2(k-1) - 1 = 2k-3$ edges, yielding $2k-3$ new trees at level k.

trees, containing $n$ leaves: 

- Rooted: $x(2n-3)$

for $n=20$ this is $10^{21}$!

- $n$ leaves
- $n-2$ internal nodes
- $2n-1$ edges

P1: induction on $n$

Adding 3rd species

Each new species adds 2 new edges

\[(2n-3)!! = \prod_{i=3}^{n}(2i-3) = 3 \cdot 5 \cdot \ldots \cdot (2n-3)\]
Exploring the Space of Trees

- we’ve considered how to find the minimum number of changes for a given tree topology

- need some search procedure for exploring the space of tree topologies

- Given $n$ sequences there are $\prod_{i=3}^{n}(2i - 3) = 3 \cdot 5 \cdot \ldots \cdot (2n - 3)$ possible rooted trees
Example Large Parsimony