Evolutionary Tree Analysis

Young-Rae Cho
Associate Professor
Department of Computer Science
Baylor University

Overview

- Backgrounds
- Distance-Based Evolutionary Tree Reconstruction
- Character-Based Evolutionary Tree Reconstruction
Phylogenetic Tree

- Phylogenetics
  - The study of evolutionary relatedness among species
- Phylogenetic Tree (Evolutionary Tree)
  - Tree-structure diagram showing the inferred evolutionary relationships between a set of objects
  - The objects are called **taxa**
    - individual genes, or
    - species when orthologous genes are used
  - Each node represents each taxon
  - Each edge represents the evolutionary relationship between taxa

Types of Phylogenetic Trees (1)

- Rooted Tree vs. Unrooted Tree
  - Rooted tree
  - Root - the most ancient species
  - External nodes (leaf nodes)
    - nodes with degree=1
    - existing species
  - Internal nodes
    - nodes with degree > 1
    - hypothetical ancestral species (before speciation events)
  - Unrooted trees
Types of Phylogenetic Trees (2)

- Cladogram vs. Additive Tree vs. Ultrametric Tree
  - Cladogram
    - Defines tree topology only
    - No meaning on branch lengths
  - Additive tree
    - Branch lengths are a measure of evolutionary divergence → evolutionary distance
    - Weighted trees
  - Ultrametric tree
    - The vertical axis is a time scale
    - Rooted trees

Types of Phylogenetic Trees (3)

- Bifurcating vs. Multifurcating
  - Bifurcating (or Dichotomous)
    - Each taxon as an internal node diverges into two separate descendent taxa
    - Fully resolved
    - What is the degree of internal nodes?
    - How many nodes a rooted bifurcating tree with N leaves has?
    - How many nodes an unrooted bifurcating tree with N leaves has?
  - Multifurcating (or Polytomous)
    - Each taxon diverges into more than two separate descendent taxa
    - Partially resolved
Types of Phylogenetic Trees (4)

- Condense Tree

Types of Phylogenetic Trees (5)

- Species Tree vs. Gene Tree
  - Species tree
    - Evolutionary relationships between species
  - Gene (or gene family) tree
    - Evolutionary relationship between homologous genes
    - Some branch points represent gene duplication events
    - Other branch points represent speciation events
Overview

- Backgrounds
- **Distance-Based Evolutionary Tree Reconstruction**
- Character-Based Evolutionary Tree Reconstruction

Evolutionary Distance

- Evolutionary Path
  - The path from the root to a leaf in a rooted tree
- Evolutionary Distance
  - Sum of weights of the shortest path between two leaf nodes in a weighted tree
- Example
  - Additive unrooted tree

![Additive unrooted tree diagram]
Evolutionary Tree Reconstruction

- Distance Matrix
  - Given $n$ species (or genes), $n \times n$ matrix $D$
  - $D_{ij} = \text{edit distance (inverse of sequence similarity)}$ between two species (or genes) $i$ and $j$

- Evolutionary Tree Reconstruction
  - Constructs an evolutionary tree that best fits the distance matrix
    - Finds an evolutionary tree such that the evolutionary distance between $i$ and $j$ in a tree $T$ equals to the edit distance between them
      i.e., $d_{ij}(T) = D_{ij}$

Formulation of Distance-based Tree Reconstruction

- Goal
  - Reconstructing an evolutionary tree from a distance matrix

- Input
  - An $n \times n$ distance matrix $D$

- Output
  - A weighted unrooted (or rooted) binary tree $T$ with $n$ leaf nodes, which best fits $D$, if $D$ is additive
Additive / Non-additive Distance Matrix

- Additive Distance Matrix
  - Matrix \( D \) is additive if there exists an evolutionary tree \( T \) such that \( d_{ij}(T) = D_{ij} \)
  - Example ??

- Non-additive Distance Matrix
  - Matrix \( D \) is non-additive otherwise
  - Example ??

Four Point Condition

- Four Point Condition
  - Given 4\(\times\)4 distance matrix \( D \) with the elements \( a, b, c, d \), among three sums of \( (D_{ac}+D_{cd}), (D_{bc}+D_{dc}), \) and \( (D_{ad}+D_{bd}) \), two are the same, and the other is smaller

- Example

- Theorem
  - An \( n\times n \) matrix \( D \) is additive if and only if the four point condition holds for every 4 distinct elements, \( 1 \leq a, b, c, d \leq n \)
UPGMA Algorithm

- UPGMA
  - Unweighted pair-group method using arithmetic averages

- Input
  - Distance matrix $D$: $n \times n$ matrix of evolutionary distance for $n$ species

- Output
  - An ultrametric (rooted) tree for $n$ species

**Process**
1. Merge two closest nodes, $x$ and $y$, to create one ancestor node $z$
2. Draw the height of $z$ by distance between $x$ and $y$
3. Remove the rows and columns of $x$ and $y$ in $D$
4. Insert the row and column of $z$ with average distance in $D$
5. Repeat (1)~(4) until reaches the root

3-Leaf Tree

- A basic component for unrooted, additive (weighted) tree for 3 species

- $n$-Leaf Tree
  - How many edges for $n$ species?
  - How many variables?
  - How many equations?

\[
\begin{align*}
   d_{jc} + d_{kc} &= D_{jk} \\
   d_{ic} + d_{kc} &= D_{ik} \\
   d_{jc} + d_{kc} &= D_{jk} \\
   d_{ic} &= (D_{ij} + D_{ik} - D_{jk}) / 2 \\
   d_{jc} &= (D_{ij} + D_{jk} - D_{ik}) / 2 \\
   d_{kc} &= (D_{ki} + D_{kj} - D_{ij}) / 2
\end{align*}
\]
Solving by Neighboring Leaves (1)

- Neighbors
  - A pair of nodes that are separated by just one other node

- Input
  - Distance matrix D: $n \times n$ matrix of evolutionary distance for $n$ species

- Output
  - An unrooted, additive (weighted) tree for $n$ species

- Process
  1. Find two closest nodes, $x$ and $y$, as neighbors
  2. Calculate distance from $x$ and $y$ to their ancestor $z$ by 3-leaf tree formula
  3. Remove the rows and columns of $x$ and $y$ in $D$
  4. Insert the row and column of $z$ with distance by 3-leaf tree formula
  5. Repeat (1)~(4) until completes an unrooted tree

Solving by Neighboring Leaves (2)

- Example
  - Distance matrix
    
    |   A | B | C | D |
    |-----|---|---|---|
    | A   | 0 | 2 | 6 | 5 |
    | B   | 2 | 0 | 6 | 5 |
    | C   | 6 | 6 | 0 | 3 |
    | D   | 5 | 5 | 3 | 0 |

- Problem
  - The closest leaves are NOT necessarily neighbors

Diagram:

```
1       2       3       4       5
|       |       |       |       |
6 ---   7 ---   8 ---   9 ---
|       |       |       |
1       2       3       4
```
Solving by Degenerate Triples (1)

- **Degenerate Triples**
  - A set of three distinct elements $1 \leq i, j, k \leq n$ where $D_{ij} + D_{jk} = D_{ik}$
  - The leaf node $j$ in a degenerate triple $i, j, k$ lies on the evolutionary path from $i$ to $k$

- **Process**
  - If distance matrix $D$ has a degenerate triple $i, j, k$, then we remove $j$ from $D$ to reduce the size of the problem
  - If distance matrix $D$ does not have a degenerate triple $i, j, k$, then we create a degenerate triple in $D$ by shortening all hanging edges in the tree

Solving by Degenerate Triples (2)

- **Shortening Hanging Edges**
  - All hanging edges are reduced by the same amount $\delta$
  - All pair-wise distances in the matrix are reduced by $2\delta$

- **Example**
  - $\delta = 1$  
  - $\delta = 3$
Overview

- Backgrounds
- Distance-Based Evolutionary Tree Reconstruction
- Character-Based Evolutionary Tree Reconstruction

Background of Character-Based Approach

- Main Concept
  - Find evolutionary history with the minimum number of character changes (or mutations) between species (i.e., ortholog genes)
  - Consider mutation at each position of the sequences separately (single point mutations)

- Edge Weight
  - Observed character differences resulted from the mutations
  - Hamming distance between two species (i.e., two ortholog genes)
Parsimony Methods

- Parsimony Score
  - Sum of the weights of all edges in the phylogenetic tree
  - Examples
    - higher parsimony score
      - less parsimonious
    - lower parsimony score
      - more parsimonious

- Evolutionary Tree Reconstruction
  - Constructs an evolutionary tree having the lowest parsimony score

Formulation of Character-based Tree Reconstruction

- Goal
  - Finding an evolutionary tree with $n$ leaf nodes

- Input
  - An $n \times m$ alignment matrix $D$
    - $n = \#$ species (sequences)
    - $m = \#$ characters (length of each sequence)

- Output
  - An evolutionary tree $T$ with $n$ leaf nodes, minimizing the parsimony score
Process of Character-based Tree Reconstruction

- Tasks
  - Assigning $n$ sequences to leaf nodes
  - Determining sequences at internal nodes

- Process
  - For each position, find the node labels (a character for each node) minimizing the parsimony score

Formulation of Small Parsimony Problem

- Goal
  - Finding the most parsimonious labeling of the internal nodes in an evolutionary tree

- Input
  - A rooted tree $T$ with $n$ leaf nodes labeled by $n$ strings of length-$m$

- Output
  - Labels (strings) of internal nodes of $T$, minimizing the parsimony score
Fitch Algorithm (1)

- Fitch Algorithm
  1. Assigns a set of characters to each node, traversing the tree from leaf nodes to root
     a. If two sets of characters from child nodes $u$ and $w$ of a node $v$ overlap, assigns the common set of them to $v$
     b. If not, assigns the combined set of them to $v$

\[ S_v = \begin{cases} 
S_u \cap S_w, & \text{if } S_u \text{ and } S_w \text{ overlap} \\
S_u \cup S_w, & \text{otherwise}
\end{cases} \]

2. Assigns labels to each node, traversing the tree from root to leaf nodes
   a. For the root, chooses one arbitrarily from its set of characters
   b. For all other nodes, if its parent’s label is in its set of characters, assigns its parent’s label
   c. Else, choose one arbitrarily from its set of characters

Fitch Algorithm (2)

- Example

- Parsimony score?
Unweighted vs. Weighted Parsimony

- **Unweighted Parsimony Problem**
  - Evolutionary tree
  - Scoring matrix

- **Weighted Parsimony Problem**
  - Scoring matrix
  - Evolutionary tree

Formulation of Small Weighted Parsimony Problem

- **Goal**
  - Finding the minimal weighted parsimony score labeling of the internal nodes in an evolutionary tree
  - Extended version of the small parsimony problem

- **Input**
  - A rooted tree \( T \) with \( n \) leaf nodes labeled by \( n \) strings of length-\( m \) having \( k \) distinct characters
  - \( k \times k \) scoring matrix

- **Output**
  - Labels (strings) of internal nodes \( T \), minimizing the weighted parsimony score
Sankoff Algorithm (1)

- Sub-tree

- Sankoff Algorithm
  - Calculating a parsimony score for every possible label at each node \( v \)
    
    \[ s_t(v) = \min \text{ the minimum parsimony score of the sub-tree rooted at } v \]
    
    if \( v \) has the character \( t \)
  - Scoring at each node based on the scores of its child nodes
    
    \[ s_t(v) = \min \left( s_t(u) + \delta_{tu} \right) + \min \left( s_t(w) + \delta_{tw} \right) \]
    
    \[ \rightarrow \text{ dynamic programming} \]

Sankoff Algorithm (2)

- Example

```
<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>
```

```
A  3  4  0  2  9  3  4  0  2  9  3  4  0  2  9
T  0  3  4  9  3  0  4  9  3  0  4  9  3  0  4  9
G  4  2  0  4  9  4  2  0  4  9  4  2  0  4  9
C  9  4  4  0  2  9  4  4  0  2  9  4  4  0  2  9
```
Formulation of Large Parsimony Problem

Goal
- Finding an evolutionary tree with \( n \) leaf nodes, having the minimal parsimony score

Input
- An \( n \times m \) alignment matrix \( D \)
  - \( n \) = # species (sequences)
  - \( m \) = # characters (length of each sequence)

Output
- An evolutionary tree with \( n \) leaf nodes labeled by \( n \) rows of length \( m \) and internal nodes labeled by strings, such that the parsimony score is minimized

Exhaustive Search Algorithm

Process
1. Enumerates all possible tree structures with \( n \) leaf nodes
2. Solves the small parsimony problem for each structure
3. Selects the best one

Problem
- Number of all possible tree structures grows exponentially w.r.t. \( n \)
Greedy Algorithm

- Nearest Neighbor Interchange Algorithm
  1. Starts with an arbitrary tree
  2. Interchanges two neighbor trees if it provides the best improvement in parsimony score
  3. Repeat (2) in each subtree

- Example

Questions?

- Lecture Slides are found on the Course Website, web.ecs.baylor.edu/faculty/cho/5330