Lecture 8, Phylogenetic 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
  - Each node represents each object → individual genes or species
  - Each edge represents the evolutionary relationship between objects

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 descendents taxa
    - Partially resolved
Types of Phylogenetic Trees (4)

- **Condense Tree**

![Condense Tree Diagram]

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

Definition of 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
Evolutionary Tree Reconstruction

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

- **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., $d_{ij}(T) = D_{ij}$
  - Value range of $d_{ij}(T)$?

---

Formulation of Distance-based Tree Reconstruction

- **Goal**
  - Reconstructing an evolutionary tree from a distance matrix for $n$ species (genes)

- **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 $(D_{ac} + D_{cd})$, $(D_{ad} + D_{bd})$, and $(D_{bc} + D_{bd})$,
    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

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

\[
\begin{align*}
  d_{ik} + d_{ij} &= D_{ij} \\
  d_{ik} + d_{jc} &= D_{jk} \\
  d_{ij} + d_{jc} &= D_{ik}
\end{align*}
\]

\[
\begin{align*}
  d_{ik} &= (D_{ij} + D_{jk} - D_{jk}) / 2 \\
  d_{ij} &= (D_{ik} + D_{jk} - D_{ik}) / 2 \\
  d_{jc} &= (D_{ij} + D_{ik} - D_{ij}) / 2
\end{align*}
\]

- **$n$-Leaf Tree**
  - How many edges for $n$ species?
  - How many variables?
  - How many equations?
Solving by Neighbor Joining (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 Neighbor Joining (2)

- Example
  - Distance matrix

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>0</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

- Problem
  - The closest leaves are NOT necessarily neighbors
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
  ```
  δ = 1
  A | B | C | D
  A | 0 | 2 | 3 | 4
  B | 2 | 0 | 5 | 6
  C | 3 | 5 | 0 | 7
  D | 4 | 6 | 7 | 0
  δ = 3
  A | B | C | D
  A | 0 | 2 | 3 | 4
  B | 2 | 0 | 5 | 6
  C | 3 | 5 | 0 | 7
  D | 4 | 6 | 7 | 0
  ```
Overview

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

Background of Character-Based Approaches

- **Main Concept**
  - Find evolutionary history with the minimum number of character changes (or mutations) between sequences (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
  - Total number of mismatches between two sequences (i.e., two ortholog genes)
Parsimony Score Calculation

- Parsimony Score
  - Sum of the weights of all edges in the phylogenetic tree
  - Examples?
    - Trees with a higher parsimony score vs. a lower parsimony score
    - Trees of less parsimonious vs. 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×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

- Process
  1. Assigning \( n \) sequences to leaf nodes
  2. Determining sequences at internal nodes

- Algorithm for Step (2)
  - For each position of the sequences in the leaf nodes, find the labels (characters) of internal nodes 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
    - If two sets of characters from child nodes $u$ and $w$ of a node $v$ overlap, assigns the common set of them to $v$
    - 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
    - For the root, chooses one arbitrarily from its set of characters
    - For all other nodes, if its parent’s label is in its set of characters, assigns its parent’s label
    - 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 $v$
  
  $s_t(v) = \min \left( s_t(u) + \delta_{tu} \right) + \min \left( s_t(w) + \delta_{tw} \right)$

  if $v$ has the character $t$

- Scoring at each node based on the scores of its child nodes

  $s_t(v) = \min \left( s_t(u) + \delta_{tu} \right) + \min \left( s_t(w) + \delta_{tw} \right)$

  $\rightarrow$ dynamic programming

Sankoff Algorithm (2)

- Example

<table>
<thead>
<tr>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<td>G</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>9</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

BAYLOR
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/3360