

Algorithms in Bioinformatics I, WS2002/3

Assignment sheet # 8

Daniel Huson

December 8, 2002

1 Distances directly obtained from ultrametric trees (1 point)

Prove that a distance matrix D that is directly obtainable from a ultrametric tree is ultrametric.

2 Newick format with edge lengths (2 points)

In the lecture we discussed the problem of reading and writing phylogenetic trees in the Newick format. However, the described algorithms only handle trees without edge lengths.

Recall that in the Newick format, each pair of matching brackets corresponds to an internal node and each taxon label corresponds to an internal node. To add edge lengths to the format, specify the length len of the edge along which we visited a given node by adding $:len$ behind the closing bracket, in the case of an internal node, and behind the label, in the case of a leaf.

For example, consider the tree $((a,b,(c,d)))$. If all edges have the same length 1, then this tree can be written as $(a:1,b:1,(c:1,d:1):1)$.

In the lecture we discussed algorithms for reading and writing trees in Newick format without edge lengths. Show how to modify both algorithms so as to accommodate edge lengths.

Implementation of a Distances class

In the four following problems, the goal is to implement a class `Distances` that can

1. determine whether the given distances are ultrametric,
2. determine whether the given distances are additive,
3. return the UPGMA tree for the given distances, and
4. return the neighbor-joining tree for the given distances.

To solve the problems below, please download the following file:

www-ab.informatik.uni-tuebingen.de/teaching/ws02/abi1/programs/program10.zip. The download contains a file called `Program10.java`. This file contains a program that reads a distance matrix from a file and then calls the different methods of the distances class.

The download contains a file called `Distances.java` that contains the framework of a distances class. Additionally, the download contains a file `PhylogeneticTree.java` that provides all necessary functionality of a phylogenetic tree. Not modification of this file is necessary.

Additionally, the download also contains all necessary `jloda` classes and their documentation. Please run the program on all supplied data sets.

The problems are solved by modifying the file `Distances.java`. The files `PhylogeneticTree` and `Program10.java` should not be modified (except for addition of your name, please).

3 Determining whether a distance matrix is ultrametric (1 point)

Implement a method `Distances.isUltrametric()` that determines whether a given distance matrix is ultrametric.

4 Determining whether a distance matrix is additive (1 point)

Implement a method `Distances.isAdditive()` that determines whether a given distance matrix is additive.

5 Implementation of UPGMA (3 points)

Implement a method `Distances.getUPGMATree()` that returns the phylogenetic tree obtained from the given distances using the UPGMA algorithm.

6 Implementation of neighbor-joining (3 points)

Implement a method `Distances.getNJTree()` that returns the phylogenetic tree obtained from the given distances using the neighbor-joining algorithm.

Additional useful activities

(a) Try to prove that the circular embedding algorithm defined in the lecture always produces a proper embedding in which no two edges intersect, not matter properties the tree has.

(b) Design an algorithm that writes rooted phylogenetic trees from left to right in the following format:

```

          +----- A
        +-----+
        |       +----- B
    +-+
    | | +----- C
    | +---+
---+ +----- E
    |
    +----- F
```

Due by 10am, Monday, 9 Dec 2002