CS 476/590 Bioinformatics Project II: Phylogenetic Trees

Description: There are two components to this project : UPGMA and Neighbor Join implementation.

(i) You are to implement the UPGMA clustering algorithm on distance matrix inputs to construct phylogenetic trees in Newick format.

(ii) You are to implement the Neighbor Joining algorithm on distance matrix inputs inputs to construct phylogenetic trees in Newick format with branch distances shown.

Specifications: There are two components to this project : UPGMA and Neighbor Join implementation. The input specs for both are the same format of distance matrix inputs.

(i) UPGMA: Your input (sample files are DM-p127.txt and DM-p139.txt) are distance matrix inputs, where the sample examples were completed in the video lectures on UPGMA. You will run the centroid-linkage based UPGMA algorithm on such distance matrix inputs as described in the video lectures and textbook chapter 11.1. Calculate average inter-cluster distances efficiently as described in videos 3 and 4 on UPGMA, without needing to look again at individual pairwise distances in original distance matrix. During program execution, your **output to the console should indicate at each step which two clusters are being merged in addition to the average distance between them. Your final UPGMA output to the console should include the phylogenetic tree in the parentheses-based Newick format.** An inefficient implementation of UPGMA is uploaded as **pysip-DM.pl** with I/O expectations consistent here.

(ii) The neighbor joining algorithm is described in slide 20 of **NeighborJoining.pptx** and corresponding video lectures. Your input s (sample files are DM-p127.txt and DM-p139.txt) are distance matrix inputs. You will run the Neighbor Joining algorithm on such distance matrix inputs. From the initial distance matrix as well as after each merge step, you are to recompute the **average distances r**, the **transition distance matrix**, and the **updated distance matrix**. You are to output these to the console upon their computation. I have uploaded a perl implementation of NeighborJoin as **oyop-DM-modGE.pl** which you are encouraged to run and test. Your output should be consistent with that program's output.

What to turn in: You must turn in a single zipped file containing your source code, a Makefile if needed for compilation, and a README file indicating how to execute your program.

Your program must be written in C/C++, Java, or Python and compile using an open source compiler on our home server home.cs.siue.edu. You must test your program on the home machine.