Programming CaseStudy

Personal Details

Name: Manvi Bengani
Institute Name: Indian Institute of Technology Kanpur
Branch: Electrical Engineering
Year: Second Year (2022-2026)

Problem Statement - 1

Input Matrix =

	0	1	2	3	4
0	0.000	0.967	1.035	1.083	0.956
1	0.967	0.000	1.129	1.116	0.966
2	1.035	1.129	0.000	0.443	1.002
3	1.083	1.116	0.443	0.000	1.072
4	0.956	0.966	1.002	1.072	0.000

PS1 folder contains the solutions of pdf files containing the dendrogram structure and the hierarchical distance matrix corresponding to the single and average linkage functions

Inference: We use the following formula for average linkage

$$d(C_1, C_2) = \Sigma\, d(p,q)/(n_1 \cdot n_2)\ \forall\ p \in C_1,\ q \in C_2$$

Problem Statement - 2

The program's file name is "*Manvi_Bengani_PS2_MAIN.py*", which implements a program that reads from a file named "inp.txt" and writes the matrix to a file named "inv.txt" in the required format with the help of np.linalg.inv() function.

Error Checking: The program checks if the inverse exists otherwise prints a statement stating that the matrix is non-invertible (singular).

Problem Statement - 3

The program's file name is "*Manvi_Bengani_PS3_MAIN.py*", which implements a program that reads from a file named "inp.txt" and writes the intermediate matrices formed during the agglomerative hierarchical clustering operation using single linkage in the specified format.

Description:
The program reads from the file "inp.txt" by calling the read_matrix function. The read_matrix function strips the lines and extracts the numbers into a 2D array or a nested list named input_matrix.
Cluster_function is a **recursive function** that is responsible for updating the matrix with intermediate matrices. It first spots the lowest available distance using the "find_min_position" function. Then using the row and column name it compares the respective rows and columns to give the least distances. The rows and columns that got combined in the cluster are then deleted and an intermediate matrix is obtained. In every cycle the intermediate matrix is appended onto the output text file using the write_matrix function.

Problem Statement - 4

The program's file name is "*Manvi_Bengani_PS4_MAIN.py*", which implements a program that reads from a file named "inp.txt" and writes the dendrogram structure to a file named "dendrogram.txt" after performing the agglomerative clustering operation.

Description:

Proceeding further from Problem Statement - 3, after storing the history of merging of clusters as the program executes, we form a directed graph of the nodes to find the final levels of all cluster nodes and generate the dendrogram.

Step 1: Graph Generation
Data Structure used: Directed Graph
The directed graph is generated by iterating through each step of the merge_history and finding the cluster which broke off from its parent cluster in the previous level by matching the common elements between the current cluster and clusters at the previous level. All clusters that remain as it is in the previous level are skipped at the current iteration.
The graph stores a link from the child to its parent (which is unique).

Step 2: Assigning each cluster a level
After generating the graph, we find the level of each cluster by counting the number of nodes on its path to root node by following the parent pointers.
The dendrogram structure is generated using the levels assigned to each level and printed appropriately

Problem Statement - 5

The program's file name is "*Manvi_Bengani_PS5_MAIN.py*", which implements a program that reads from a file named "inp.txt" and writes the hierarchical distance matrix to a file named "hDist.txt" after performing the agglomerative clustering operation.

Description:

Proceeding further from Problem Statement - 4, after forming the graph, we find the LCA for all leaf nodes (pairwise) of the graph, and the corresponding clusters responsible for its formation. Based on this, we create a hierarchical matrix storing the minimum of all distances amongst the pairs from either of the clusters

Step 1: Graph Generation
Data Structure used: Directed Graph
In addition to the graph generated in PS4, we generate another directed graph, this time pointing from parents to both of its children

Step 2: Finding the LCA for each leaf node
We generate the path from each leaf node to the root node. We move along both the paths simultaneously back to leaf nodes breaking at the point where the paths diverge giving the LCA of the leaf nodes.

Example: Calling LCA on (1,6) gives two paths -
Say the paths are $[1, 2, 3, Root]$ and $[6, 5, 3, \ Root]$
Traversing the lists in a reversed manner would give:

$$Root \ = \ Root \leftrightarrow 3 \ = \ 3 \leftrightarrow 2 \ \neq \ 6$$

This shows 3 as the LCA of (1,6)

Step 3: Generating the hierarchical matrix

$$d_H(C_1, \ C_2) \ = \ min\{(u, v) \ \forall \ u \in C_1, \ v \in C_2\}$$

We generate the values of the hierarchical matrix using the above equation and print it in the specified format.