**Report**

**Introduction: -**

Clustering refers to the grouping of the items on the basis of the similarity or the correlation between them. The aim of the project was to form clusters of the given graph on the basis of density and cluster property.

**Explanation of the Source Code: -**

In the beginning, first and foremost thing was the numbering of the nodes in the given dataset. This was done by an individual program “Numbering Nodes”. The basic concept that was used here was to assign a unique corresponding number to every node once it is done, make numbered copy of the dataset. This was done by the implementation of linked list.

Now, coming to the main program, two structs are declared named “**Nodes**” and “**Edges**” which is self-explanatory, it laid out the foundation for forming the graph. The Edges consists of two values: Val and Weight, where val shows the connection between two nodes and the weight is the weight of the edge. Now, coming to the Node, it consists of the cluster, index, weight and degree. The cluster variable is of bool type which indicates whether the given node is a part of the cluster or not. The following are the functions that are used repeatedly in the program for different purposes:

1. *getNoOfNodes (returns the number of nodes in the graph)*
2. *getNoOfEdges (returns the total number of edges in the graph)*
3. *density (returns the calculated density of the cluster)*
4. *clusterProperty (return the calculated cluster property)*
5. *sortPriorityW (sorts the nodes in descending order on the basis of their weights)*
6. *sortPriorityD (sorts the nodes in descending order on the basis of their degrees)*
7. *maxWeight (returns max weight of the node in the graph)*
8. *maxDegree (returns max degree of the graph)*

Moving further, there are declarations of two matrices **G** and **G’**, similarly **Nodes** and **Nodes’** for the **graph** and **cluster**, respectively. That are initially NULL. The values are assigned to the G and nodes by reading the numbered dataset file. The dataset that is used for the testing is “**Temp**” and “**Temp1**” files, along with their images for better visualization. After that a do while loop is used to make clusters as long as there is no node in the graph. This is done by first calculating the weight of the edges and the weight of the nodes, storing their values in their respective nodes and edges. After that, a temp array is declared for storing the nodes so that they can be sorted to make clusters on the basis of the *priority.* The clustering is done on **two basis**: first on the basis of **highest weight**, if the highest weight reduces to zero then on the basis of **highest degree**. A nested for loop is used for the clustering, the simple concept that is applied here is that clustering starts from the highest weight/degree node, add nodes to the cluster as long as density and cp is greater that threshold density and cp i.e d’ and cp’. **Once all the neighbors are done, then it moves the 2nd highest weight/degree nodes that is already part of the cluster and tries to find its neighboring nodes to add into cluster as long as the density and cluster property conditions are met**. Once, the cluster is formed, it removes all the nodes and connections form the actual graph just as instructed **G=G-Cluster.** The cluster data is stored on the file with its respective number. All the formed clusters data is stored in the folder *Clusters.* Then, the clusters nodes’ and matrix G’ are initialized for the formation of the next cluster. Before forming the next cluster, the weight and common neighbors are calculated again in order to remove any errors prior to the formation of the next cluster.

**Note:** Before running the program, it should be assured that the clusters folder is empty because it may cause redundancy. Thus, producing an error. In order to use another dataset, the file name in main(), getNoOfNodes() and getNoOfEdges() function needs to be updated.