.

**RFunctions :**

Following functions were used in project :

Decompose.graph(astrocollab)

graph2=simplify(ast1,remove.multiple = TRUE,remove.loops=TRUE);is\_simple()

neighbours()

degree(astrocollab)

V(graph)

as\_adjacency\_matrix(astrocollab)

as\_adj\_list(astrocollab)

neighbourhood.size()

plot(astrocollab)

evcent()

t(adjmatrix)

power\_centrality()

clusters(astroNew)

upgrade\_graph()

vertex\_attr()

is.simple()

library(igraph)

vcount(ast2)

ecount(ast2)

which(V(astroDemand[[7]]) $name == "$a3[[1]]");

as\_simple\_path(ast2,1,2)

is\_simple(ast2)

max(degree(ast2))

ego(ast2,order=1)

clique(ast2)

clique\_num(ast2)

betweenness()

ego(ast2)

ego\_size(ast2)

**Simplification:**

Decomposed graph into connected components and eliminated connected components having number of vertices less than 3.

Program :

library("igraph");

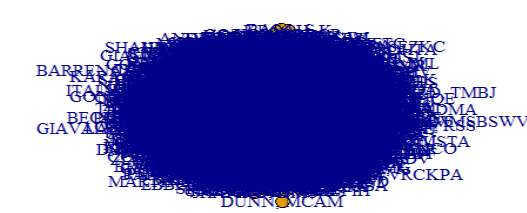
setwd("C:/Users/Om/Desktop/prev/Spring 2016/Big data/Project");

load("astrocollab.Rdata");

ast1 <- upgrade\_graph(astrocollab);

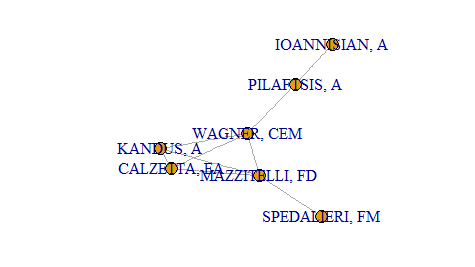
ast2 <- simplify(ast1, remove.multiple = TRUE, remove.loops = TRUE,edge.attr.comb = igraph\_opt("edge.attr.comb"));

plot(ast2) :



all\_clusters <- decompose.graph(ast2, mode = c("weak", "strong"), max.comps = NA,min.vertices = 3);

plot(all\_clusters[[10]], layout=layout.fruchterman.reingold)



|  |
| --- |
|  |
| **Usage of functions of R :**  1. decompose() :      2. plot() :  To plot the graph :      3. delete.vertices() :  To delete vertices with degree less than 2 , following command was executed :        4. as\_adjacency\_matrix();  To find out adjacency matrix, following command was executed :    5. degree() :  To find degree of each vertex, following command was executed:    6. as\_adj\_list() :  To find adjacency list for graph :    7. ecount() :  To find number of edges in graph :    8. vcount() :  To find number of vertices in graph :      9. diameter() :  To find out distances table,    10. hist() :  To find Histogram,      11. neighbours() :  To find out neighbours of any vertex :    12 is\_connect() :  To check whether graph is connected or not.    13. power\_centrality() :    14. clique\_num() :  To find maximum clique ,    15. ego\_size() :  To find ego of all vertices.    16. V() :  To find all vertices of the graph.    17. which(V(astrocollab) $name == "MEEGAN, CA");    Answers for 5:  **1. Central Person:**  Person with maximum degree was evaluated.    Answer:    **2. Longest path:**  For longest path, we removed loops, circuits first. Then we decomposed graph into clusters. And all possible path lengths were calculated and maxiumum path was selected based on highest number of vertices in path.  **Code and output screenshot:**  **Graph for cluster with longest path:**    **3. Largest Clique :**  Direct Function clique\_num was used to find out maximum clique.  clique\_num(ast1);  Answer :  57.  **4. ego :**      **5. Power Centrality :**  Power\_centrality() function was giving error as    Thus, we calculated power centrality of each node using following logic :  1. Calculate Adjacency matrix  2. Calculate degree matrix.  3. Multiply both matrices to find out power centrality.  4. Print person with maximum power centrality  Program :  pc=function(ast2)  {  mat1=as\_adjacency\_matrix(ast2);  mat2=degree(ast2);  a4=mat1%\*%mat2;  a5=max(a4);  }  Answer:  Maximum power centrality = 23262  Person = FRONTERA, F  360 |