EE232E Graphs and Network Flows

Spring 2017_Project1

Team Member:
Xiongfeng Hu (304753117)
Younan Liang (504759929)
Dui Lin (504759948)

1. Introduction

In this project we study social networking and graphs of user's personal friendship network. We have tried to interpret community structures in the friendship network and their various graphical applications. Real social networks of Google + and Facebook have been used through the project. The solutions for each problems are as follows.

2. Problems

2.1 Problem 1

Using the function of *is connected ()* and *diameter ()*, we find that the network is connected, and the diameter is 8. And the degree distribution is plotted as below:

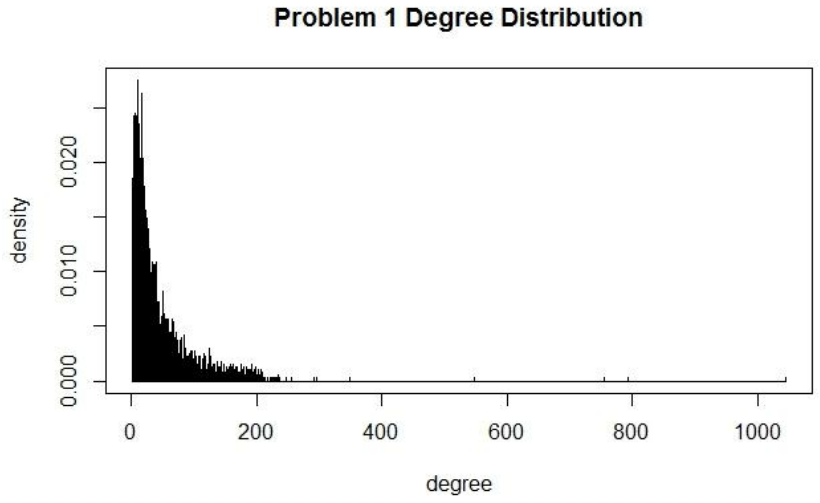

Figure 2.1.1 Degree Distribution

By looking at the degree distribution, we choose a curve to fit the degree distribution, using the function of R package. Because the question does not ask to write the function of this curve, and the function of the curve can be quite complicated, we only provide the curve's mean squared error at this time.

Choosing 2 methods to fit the curve. And 2 curves are plotted as below:

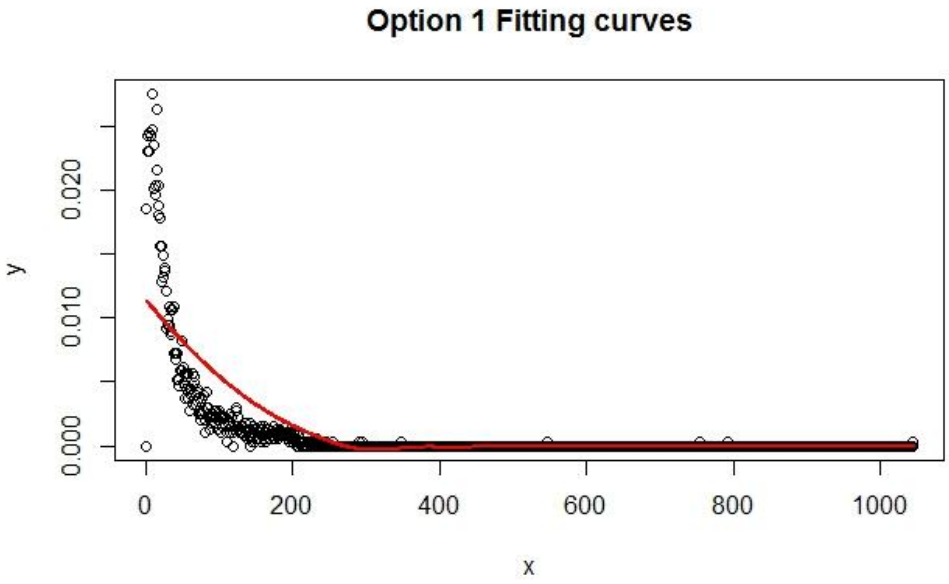

Figure 2.1.2 Degree Distribution, Option1

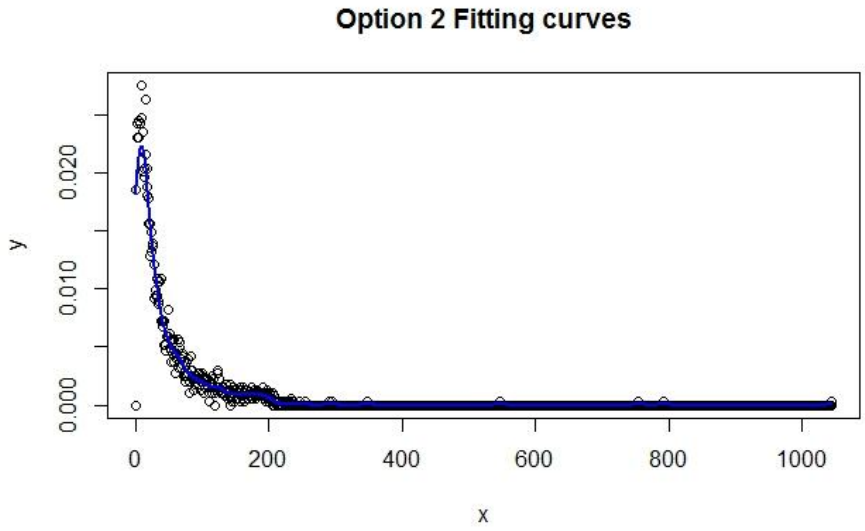

Figure 2.1.3 Degree Distribution, Option2

For option 1, the curve's total mean squared error is 3.8947e-6. For option 2, the curve's total mean squared error is 2.8950e-7. Of course the option 2 is better. With function *mean (degree (network))*, we find the average degree is 43.69.

2.2 Problem 2

With function neighbors (network, node), we find the neighbors of node 1. The graph which consists of node 1 and its neighbors and the edges that have both ends within this sets of nodes are plotted as below:

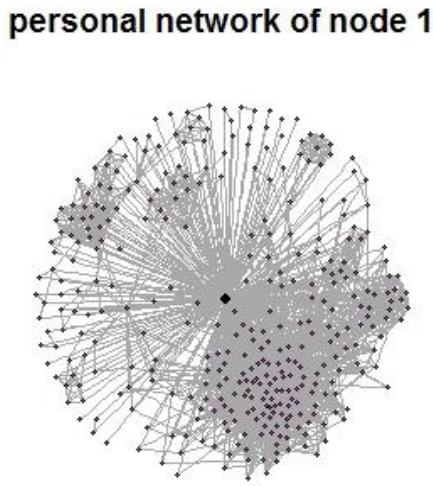

Figure 2.2 Personal network of node 1

And this graph has 348 nodes and 2866 edges.

2.3 Problem 3

By scanning each node and comparing its degree with 200, we can find all core nodes. Using the function of *degree ()* and *mean (degree [core_nodes])*, we find there are 40 core nodes in the network, and their average degree is 279.4.

Choose one core nodes for further analysis. The core node number 1 is chosen. Using different algorithms (Fast-Greedy, Edge-Betweenness, and Infomap Community), we plot these networks. Also, to measure the connection inside the community structure, the community distribution of these networks are also plotted. The community structure and

its distribution of this core node is plotted as below:

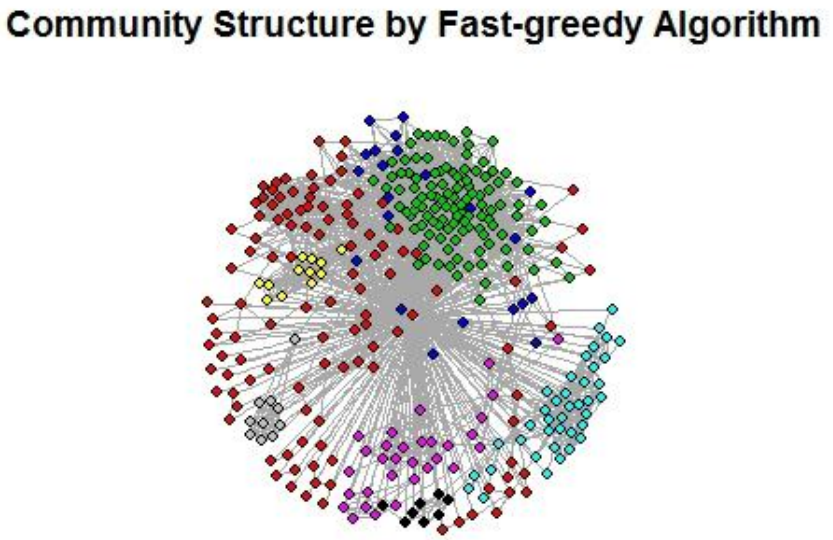

Figure 2.3.1 Community Structure Generated by Fast-greedy Algorithm

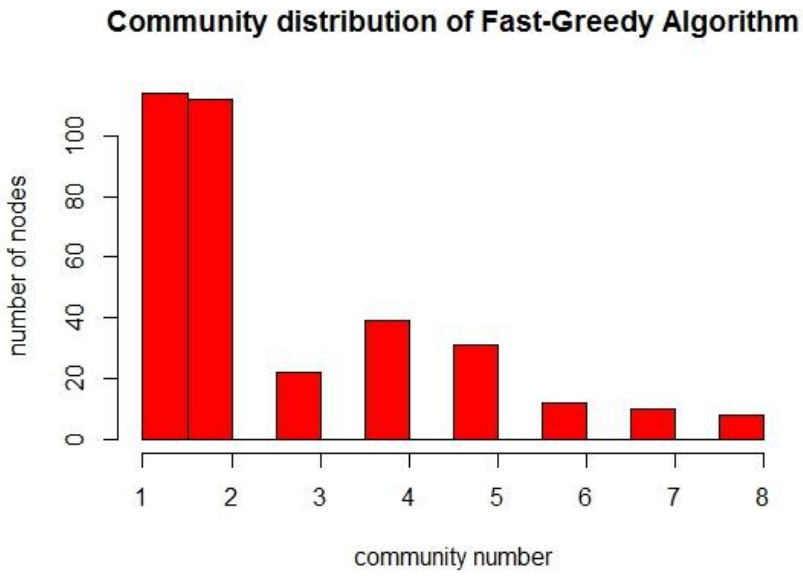

Figure 2.3.2 Community distribution of Fast-greedy Algorithm

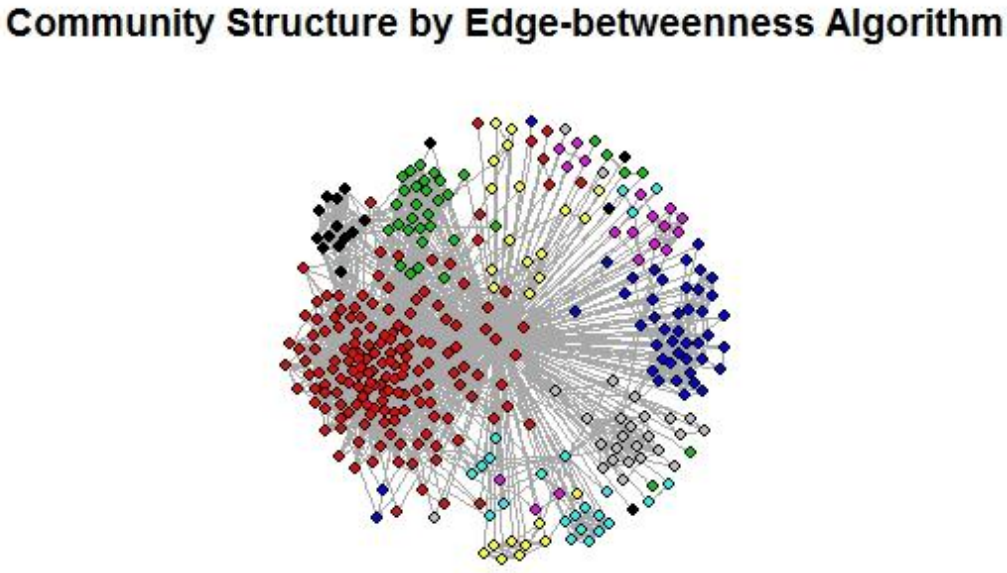

Figure 2.3.3 Community Structure by Edge-betweenness Algorithm

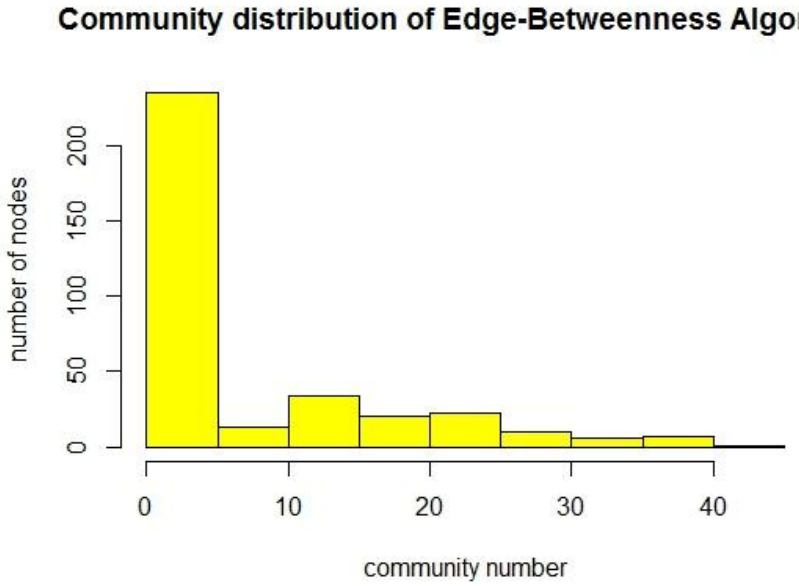

Figure 2.3.4 Community distribution of Edge-betweenness Algorithm

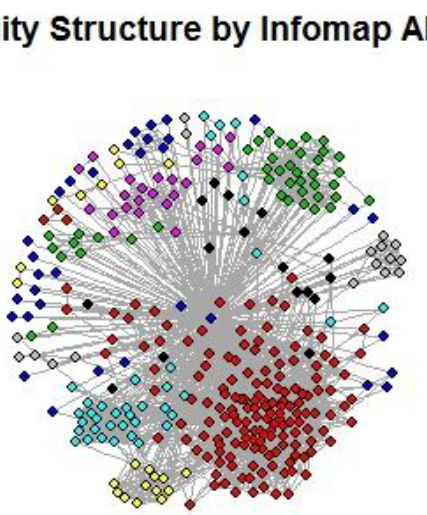

Figure 2.3.5 Community Structure Generated by Infomap Algorithm

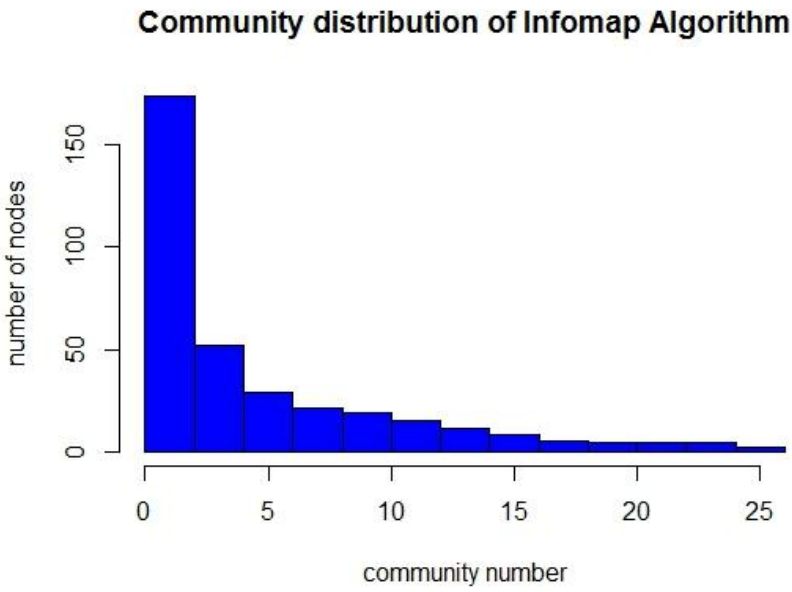

Figure 2.3.6 Community distribution of Infomap Algorithm

Analysis of the results:

1. The community structure generated by 3 algorithms are mostly similar, but they

have different features, clearly.

2. Edge-betweenness algorithm breaks the personal network with more communities, compared to 2 other algorithms. It can also be concluded with the community distribution graph. Edge-betweenness algorithm generates lots of communities who have only few nodes.
3. There is a large overlap area in these communities generated by 3 algorithms, and it can be clearly seen that the core node is in the community who has the maximum number of nodes.

2.4 Problem 4

We first generated a graph without the core node 1 and then plotted the community structure and community distribution as we did above using three different algorithms - fast-greedy algorithm, edge-betweenness algorithm and infomap algorithm. The results are shown in figure 4.1-4.6.

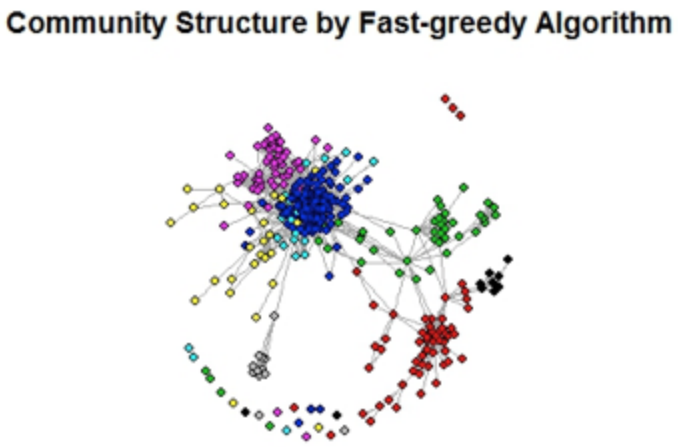

Figure 4.1 Community Structure using fast-greedy algorithm

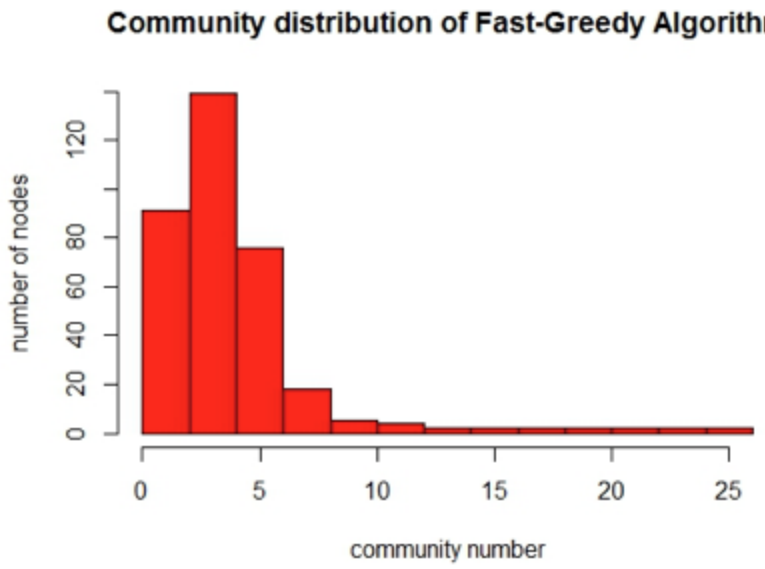

Figure 4.2 Community Distribution using fast-greedy algorithm

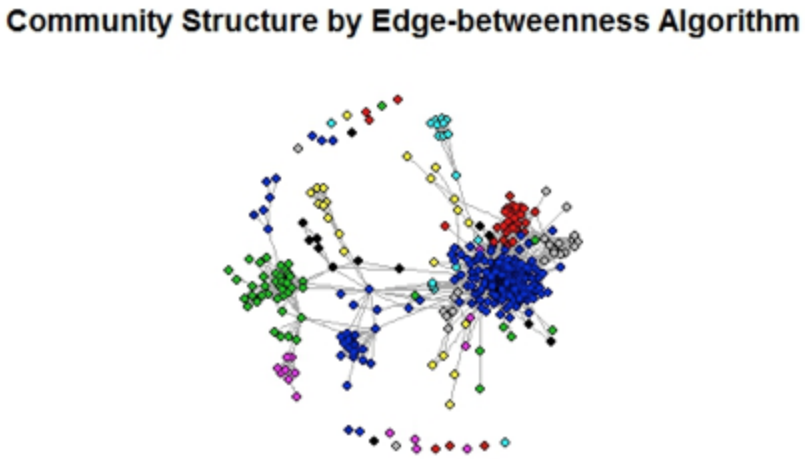

Figure 4.3 Community Structure using edge-betweenness algorithm

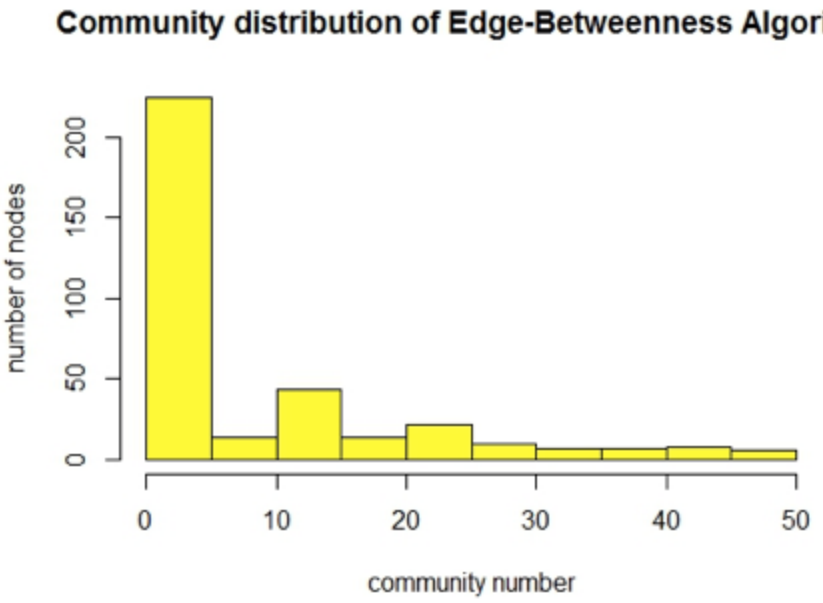

Figure 4.4 Community Distribution using edge-betweenness algorithm

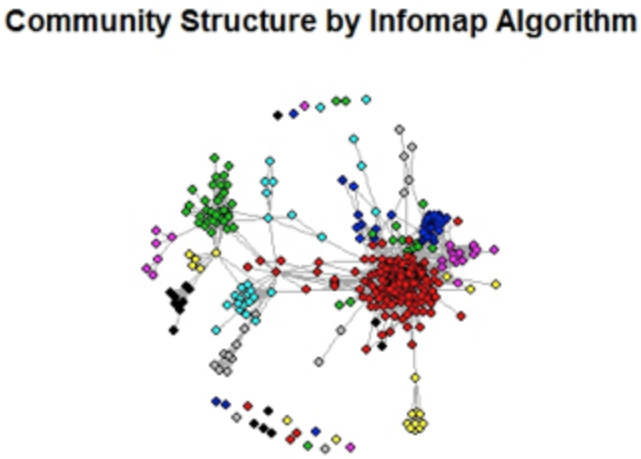

Figure 4.5 Community Structure using infomap algorithm

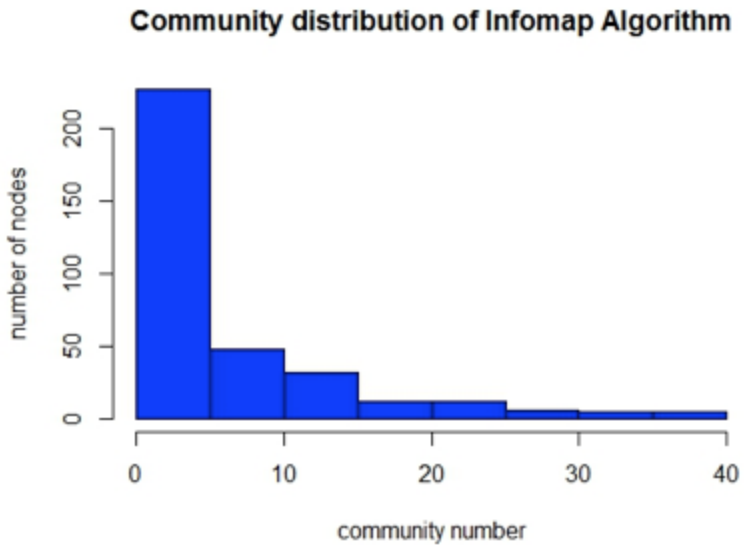

Figure 4.6 Community Structure using infomap algorithm

If we look further into those community structures, we may found that though they are structured without the core node, the partitions are actually similar. And this can be testified by examine the modularity of structures of problem 3 and problem 4. The difference of modularity is about 10% between two parts.

2.5 Problem 5

Checking the dispersion of *dispersion* and *Embeddedness*:

Embeddedness is the number of mutual friends a node shares with the core node. And larger Embeddedness means that there are more mutual friends a node have. Therefore, we can use function *intersect ()* to count the number of mutual friends of a node with a core node, and to get the Embeddedness value.

Dispersion is the sum of distances between every pair of the mutual friends a node shares with a core node. It is a value to measure the relationship of mutual friends of a node. Larger dispersion means that a node's mutual friends are familiar with each other. Therefore, we can use function *comb ()* and *shortest.paths ()* to measure

the dispersion of a node.
And the distribution of embeddedness and dispersion are plotted as below:

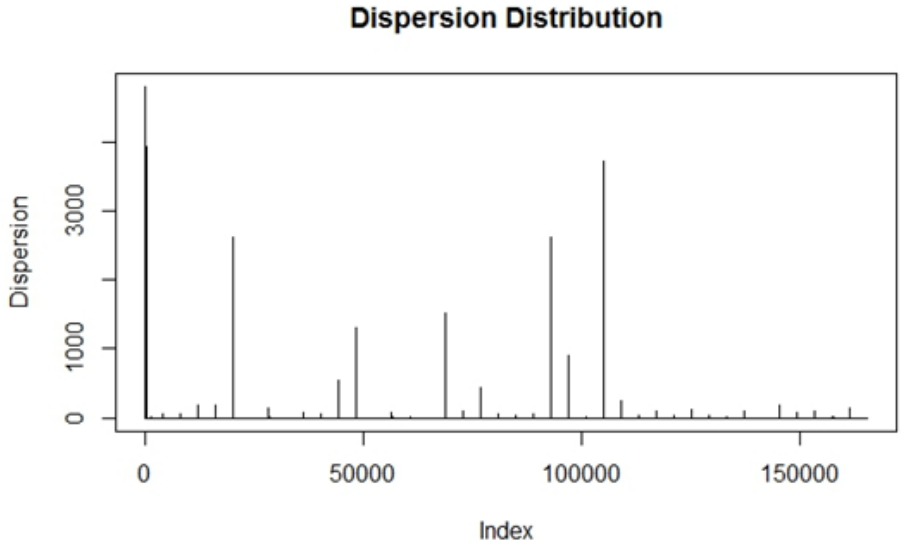

Figure 2.5.1 Dispersion Distribution

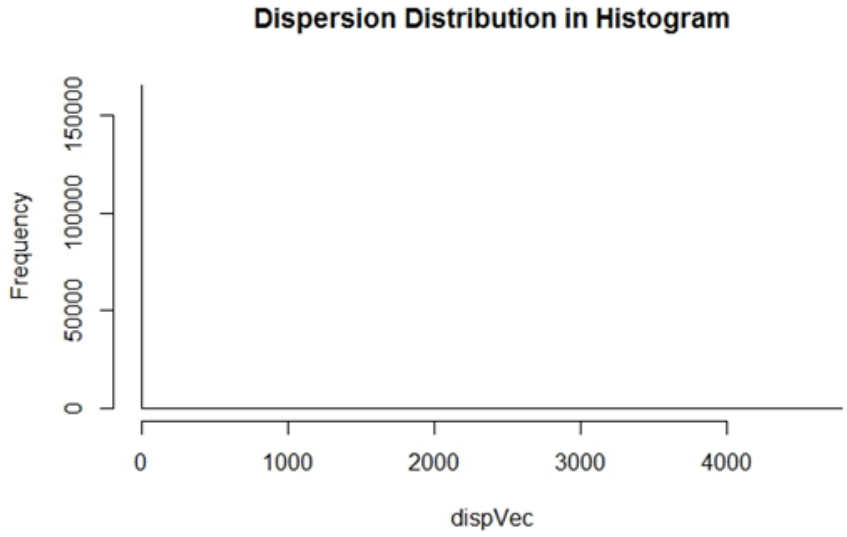

Figure 2.5.2 Dispersion Distribution in Histogram

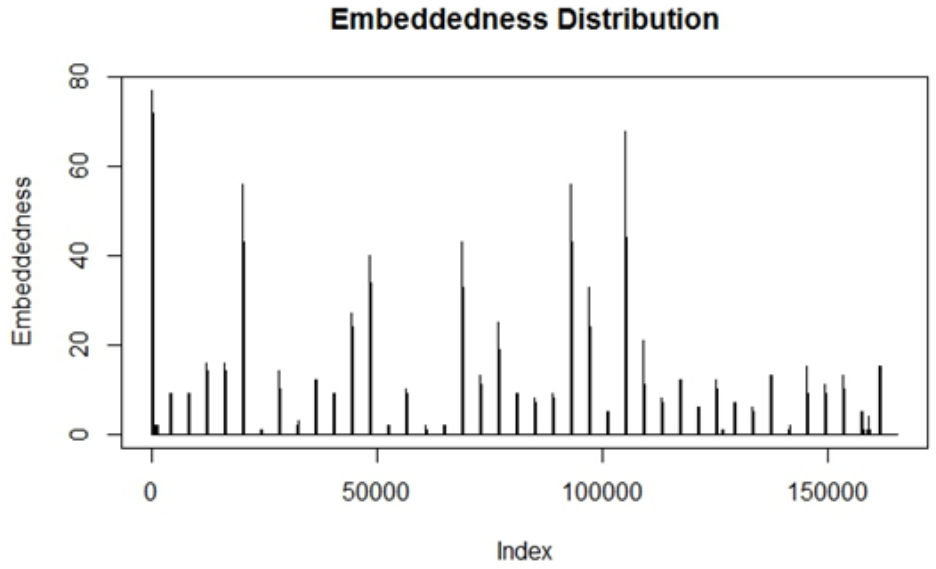

Figure 2.5.3 Embeddedness Distribution

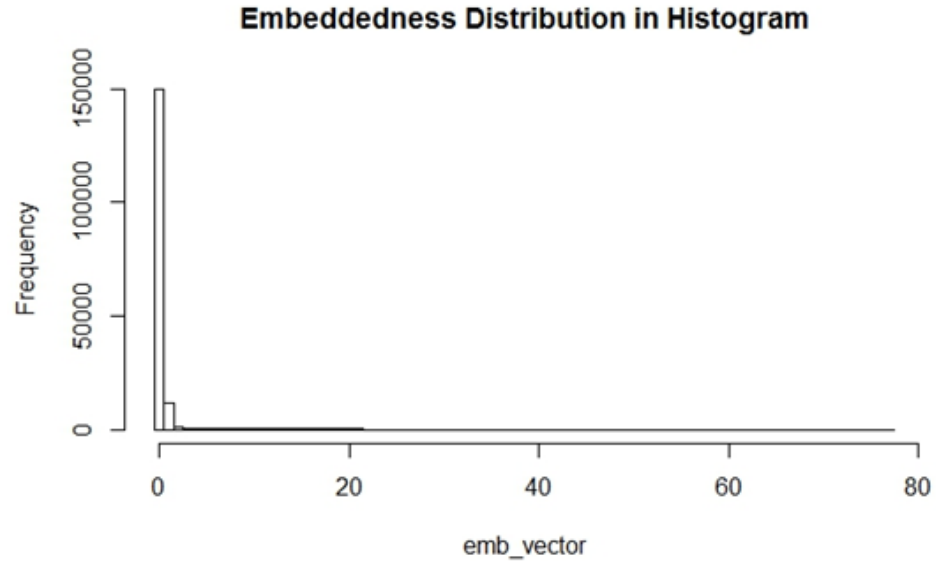

Figure 2.5.4 Embeddedness Distribution in Histogram

Then select 3 personal networks (node 1, 5 and 10), and plotted the maximum dispersion network, maximum embeddedness network, and maximum dispersion/embeddedness network, respectively. The nodes with maximum parameter and the edges incident to this node are highlighted.

Node 1:

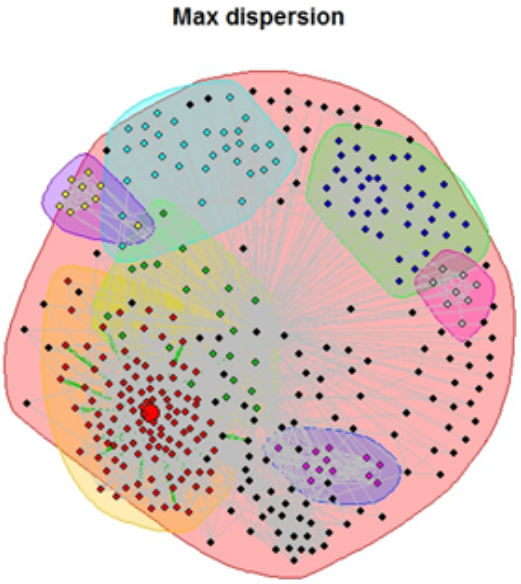

Figure 2.5.5 Max dispersion network of Node 1

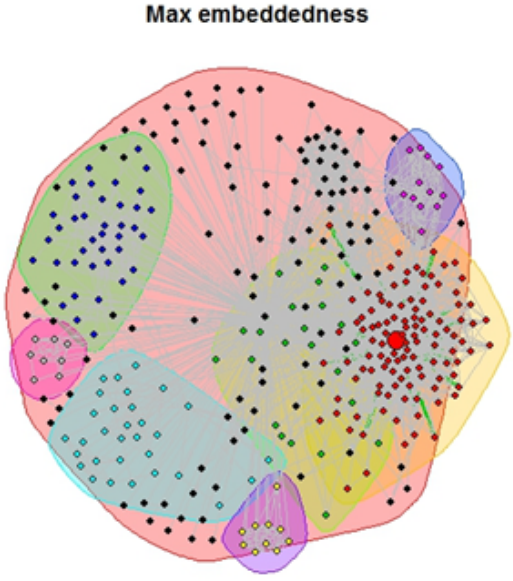

Figure 2.5.6 Max embeddedness network of Node 1

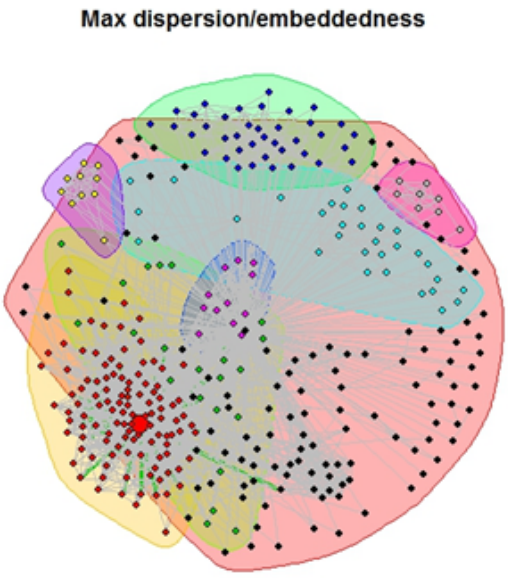

Figure 2.5.7 Max dispersion/embeddedness network of Node 1

Node 5:

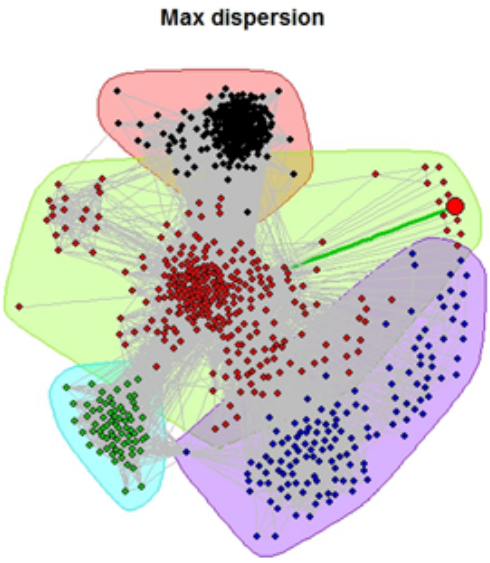

Figure 2.5.8 Max dispersion network of Node 5

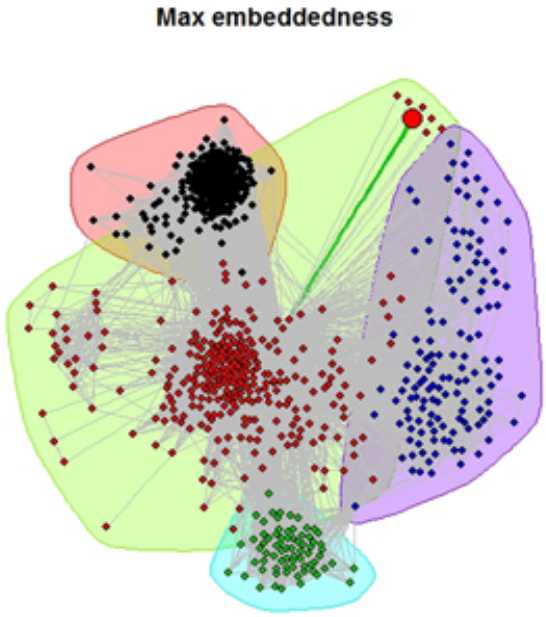

Figure 2.5.9 Max embeddedness network of Node 5

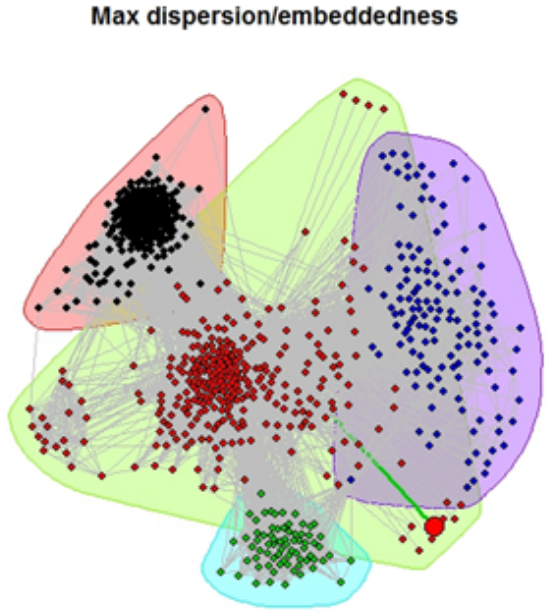

Figure 2.5.10 Max dispersion/embeddedness network of Node 5

Node 10:

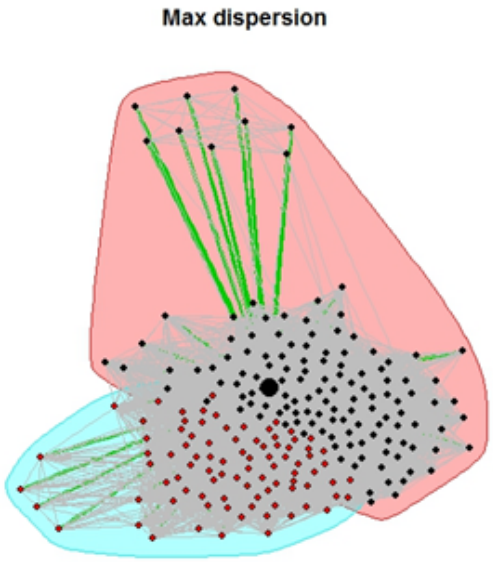

Figure 2.5.11 Max dispersion network of Node 10

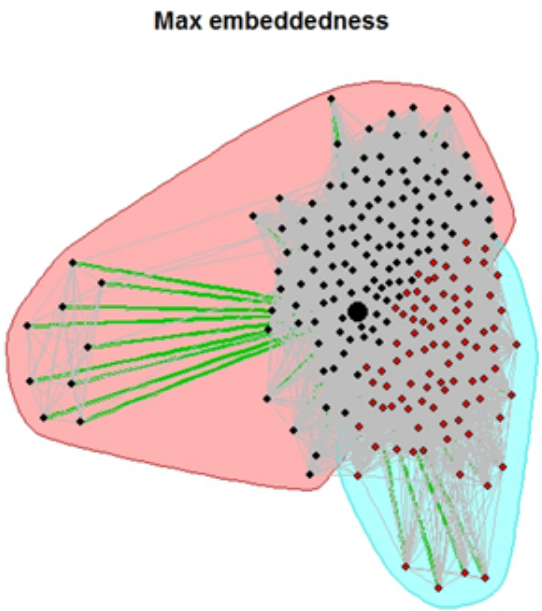

Figure 2.5.12 Max embeddedness network of Node 10

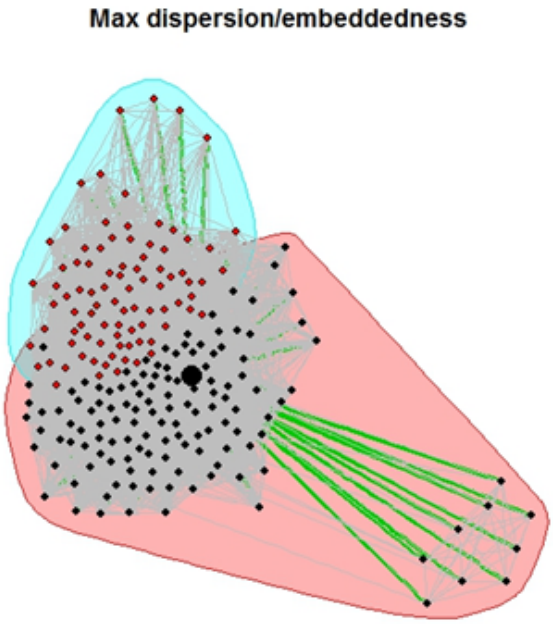

Figure 2.5.13 Max dispersion/embeddedness network of Node 10

Analysis of the results:

1. The highlighted node in Figure 2.5.5, 2.5.8, and 2.5.10, have the maximum dispersion value with its core node. From the graph or the definition of dispersion, we conclude that the mutual friends of the core node and the maximum dispersion node is least likely to know each other. For example, a person and the friend of the couple of this person.
2. The highlighted node in Figure 2.5.6, 2.5.9, and 2.5.12, have the maximum embeddedness value with its core node. From the graph or the definition of embeddedness, we conclude that the node with maximum embeddedness value shares the most mutual friends with its core node. For example, two people in the same class.
3. The highlighted node in Figure 2.5.7, 2.5.10, and 2.5.13, have the maximum dispersion/embeddedness value with its core node. From the graph or the definition of dispersion/embeddedness, we can conclude that the node's dispersion is relatively large, and the node's embeddedness is relatively small.

We may think that two communities are only linked by the core node and the maximum dispersion/embeddedness node. So the core node and the maximum dispersion/embeddedness know each other in a private situation.

4. The previous 3 cases happen quite frequently in our real life.

2.6 Problem 6

In graph theory, the conductance of graph G=(V,E) measures how "well-knit" the graph is. The conductance of a cut (S, $\bar{S}$) in a graph is defined as:

$$\varphi(S) = \frac{\sum_{i \in S, j \in \bar{S}} a_{ij}}{\min(a(S), a(\bar{S}))}$$

where the aij are the entries of the adjacency matrix for G, so that

$$a(S) = \sum_{i \in S} \sum_{j \in V} a_{ij}$$

is the total number of the edges incident with S. The conductance of the whole graph is the minimum conductance over all the possible cuts.

In this problem, we use conductance to provide a measure for the structural features of the cut (S, $\bar{S}$). Let v be the sum of degrees of nodes in community S, and s be the number of edges with one endpoint in S and the other in $\bar{S}$, which is the complement of S. Then the conductance of S is

$$\varphi(S) = \frac{s}{v} = \frac{v - 2e}{v}$$

where e is the number of edges with both endpoints in S. Here communities are considered as groups of nodes with more intra-connections than inter-connections. Therefore, we prefer communities with small conductance, which is equivalent to densely-linked inside. Here we denote two types of each network.

$$type_1 = \min(\varphi(S))$$
$$type_2 = \max(\varphi(S))$$

Type_1 communities have minimum conductance, with dense links inside and sparse links outside. On the other hand, type_2 have maximum conductance with denser links to outside. They are obtained using fast-greedy algorithm,

type_1
4 2 2 3 1 1 2 2 3 1 2 3 1 3 1 2 2 2 3 1 2 1 2 3 1 3 1 3 2 2 1 1 1 2 3 2 2 2 2 3 2
type_2
3 4 1 2 4 2 1 1 1 2 1 1 2 2 2 3 1 3 2 2 6 2 3 2 2 2 2 2 3 1 3 2 2 1 1 1 1 1 1 1 1 The number of 1-6 means the index of community in each core node's personal network

2.7 Problem 7

We now run the same kind of analysis on another real social network with tagged relationships. Google+ , unlike Facebook, has a directed network structure, where you can have someone in your circles regardless of whether they have you in their circles or they don't. Circles are tags you put on your relationships when you add people. After extract all the personal network with its circles, we choose one to show its community structure using both Walktrap and Infomap algorithms. The results are shown as below:

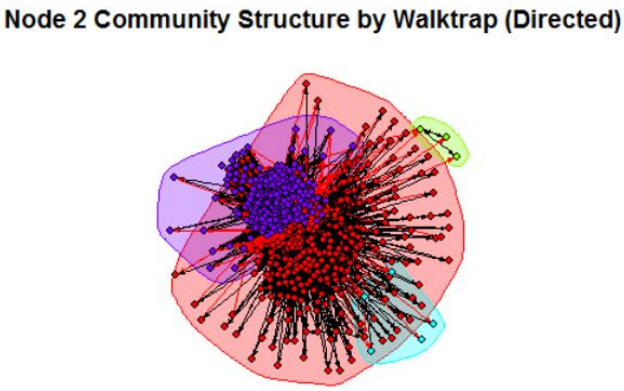

Figure 2.7.1 Community Structure by Walktrap (Directed)

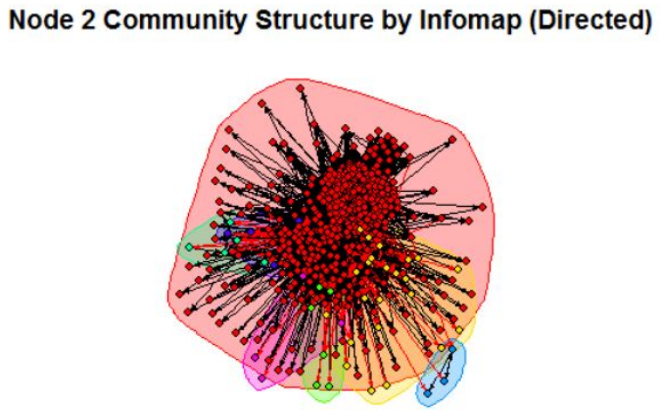

Figure 2.7.2 Community Structure by Infomap (Directed)

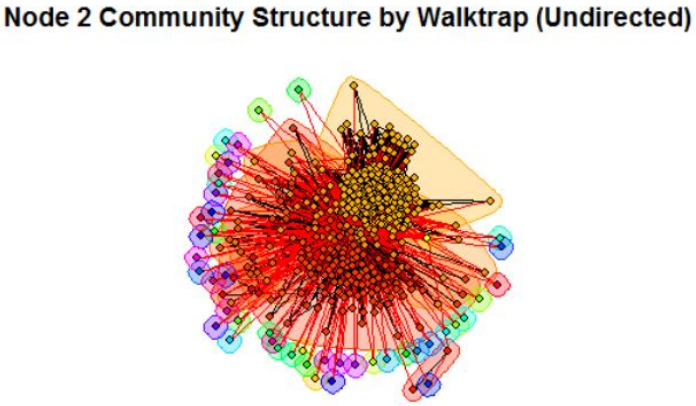

Figure 2.7.3 Community Structure by Walktrap (Undirected)

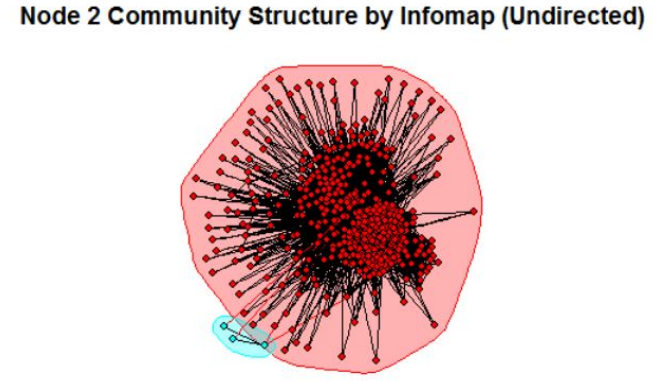

Figure 2.7.4 Community Structure by Infomap (undirected)

With further analysis, if we can tag the relationships more clearly and reduce the overlap between different circles, we can get better performance given by community detection algorithms.