Project 1 Report for EE232E

Lei Ding
Yi Ding
Sonu Mishra

Objective

In this project we will study a social network and graphs of users' personal friendship network. We mainly consider Facebook and Google+ datsets. We will explore community structures in the friendship network and their interpretation and applications. The paper[1] that we refer to in this project provides some interesting ideas on how we can find romantic relationship between nodes in such networks.

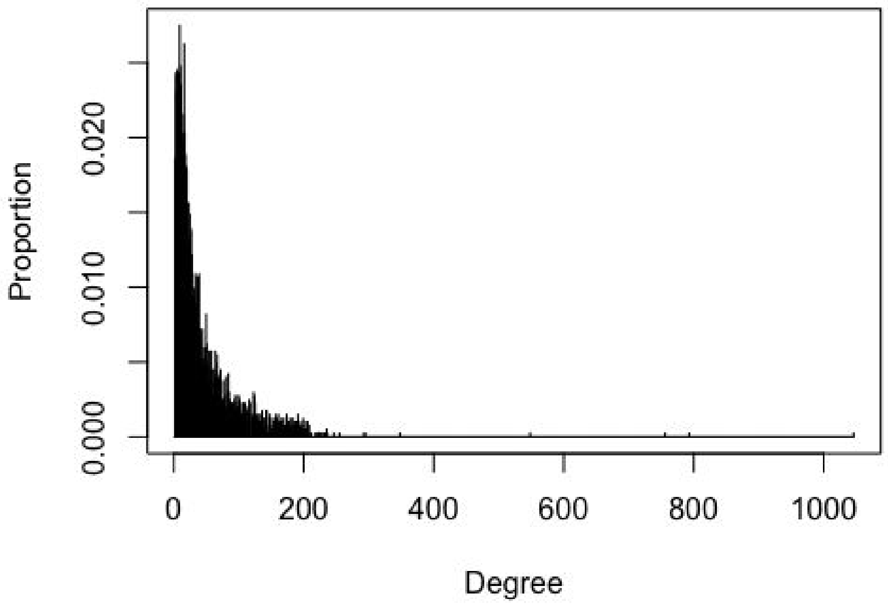
Question 1

Download the Facebook graph edgelist file facebook_combined.txt 2. Is the network connected? Measure the diameter of the network. Plot the degree distribution and try to fit a curve on it. What is your curve's total mean squared error? What is the average degree?

Solution

After briefly analyzing the network in the provided file, we can get that:

1. The network is connected.
2. The diameter of the network is 8.
3. The degree distribution is shown in Figure 1.2. And Figure 1.1 shows our different tries to fit the distribution figure. We compare five different fit method to fit this degree distribution which include y = ax + b, y = a/x, y = a + b*logx, y = a/x + b*x and y = exp(a + b*x)[1]. In this figure, we can see that the violet color curve fits the distribution best. It is the form of *y = exp(a+bx)*. We use the *ggplot2* library to realize this fit.
4. The total mean square error is 1.0652*e*-6. We can see from this mse that the curve fits the distribution figure very well.
5. The average degree is 43.7.



[1] Refer to http://stackoverflow.com/questions/14190883/fitting-a-curve-to-specific-data

Figure 1.1 Degree distribution

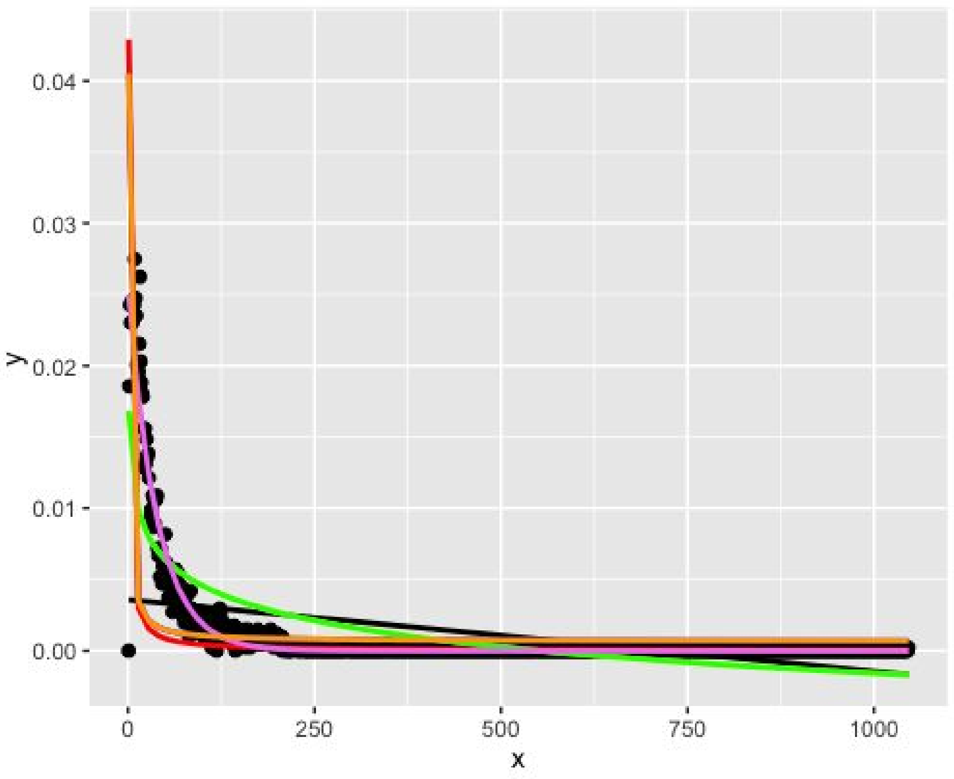


Figure 1.2 Curve fit

Question 2

Take the first node in the graph (The node whose ID is 1) and find its neighbors. Create a graph that consists of node 1 and its neighbors and the edges that have both ends within this set of nodes. We will call this the personal network of node 1. Note that the common characteristic among the nodes in this graph, except for node 1, is that they are all friends of node 1, or equivalently, node 1 is a mutual friend of all of them. How many nodes and edges does this graph have?

Solution

According to the question, we use the *induced.subgraph* function to extract the subgraph. The subgraph (personal network) for USER 1 is shown in Figure 2.1. And by *vcount* and *ecount* in R, we caculate the number of vertexes and the number of edges.

1. The number of vertexes: 348.
2. The number of edges: 2866

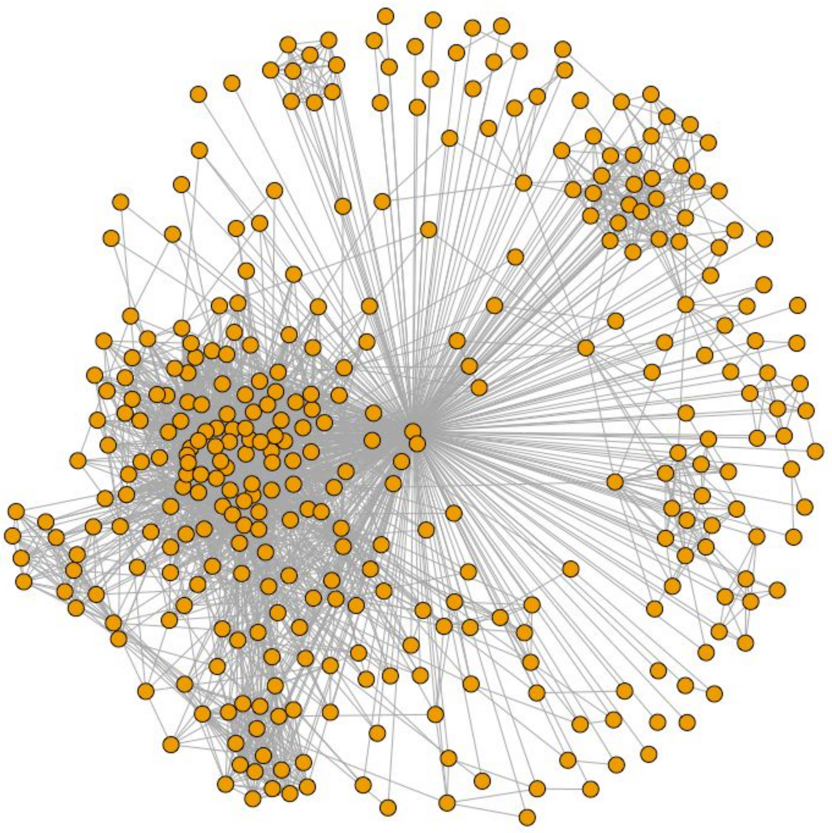


Figure 2.1 Personal network for USER 1

Question 3

Find nodes in the graph that have more than 200 neighbors. How many core nodes do you find in the network? What is the average degree of these core nodes? For one of these nodes, find the community structure of the core's personal network. Plot the network and try to distinguish communities with color. Use Fast-Greedy, Edge-Betweenness, and Infomap community detection algorithms in igraph and compare results.

Solution

For this question, we traverse the whole personal networks and try to find those whose number of neighbours are larger than 200. The answers to the sub-questions are as below:

1. The number of the core nodes is 40.
2. The average degree of these core nodes is 279.4.
3. We choose the user whose Id is 3 among these nodes as an example. We use three different community detection algorithms to explore the community structure of this node's personal network, i.e, *Fast-Greedy*, *Edge-Betweennss*, and *Infomap Community*. The community figures of these three algorithms are shown in Figure 3.1~3.3 for three algorithms respectively. We use two different plot methods to make the community clearer. And the structural features of this personal network for three algorithms are shown in Table 3.1.

Table 3.1 Structural features using three algorithms

Algorithm	Fast-Greedy	Edge-Betweenness	Infomap
Modularity	0.2503461	0.133528	0.0954642

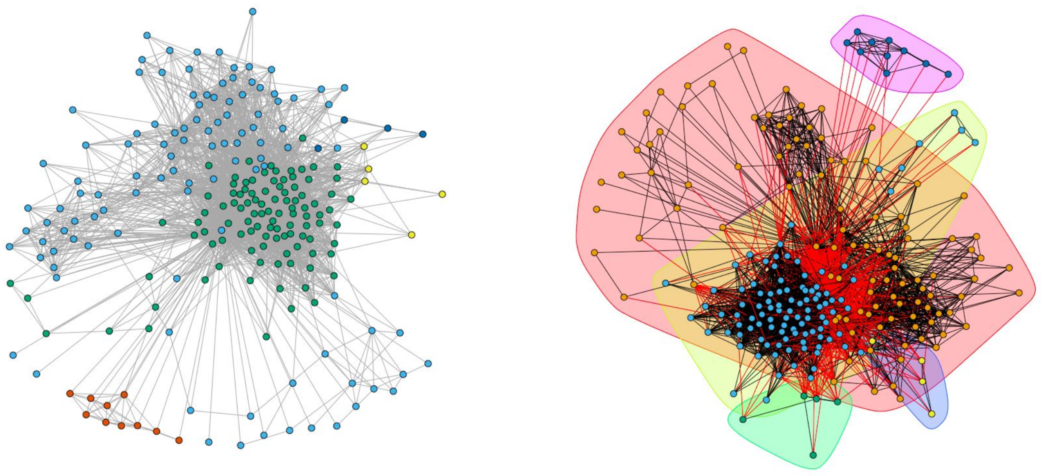


Figure 3.1 Community structure using Fast-Greedy

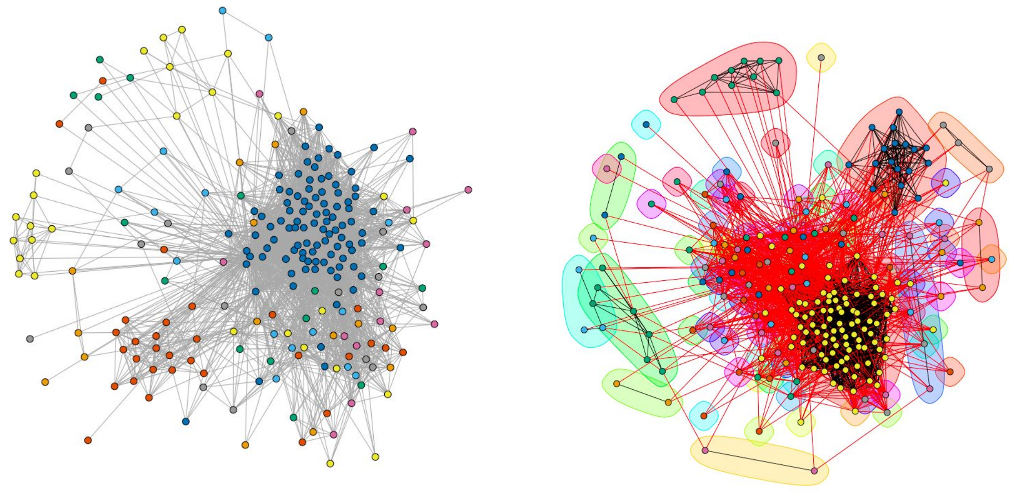


Figure 3.2 Community structure using Edge-Betweenness

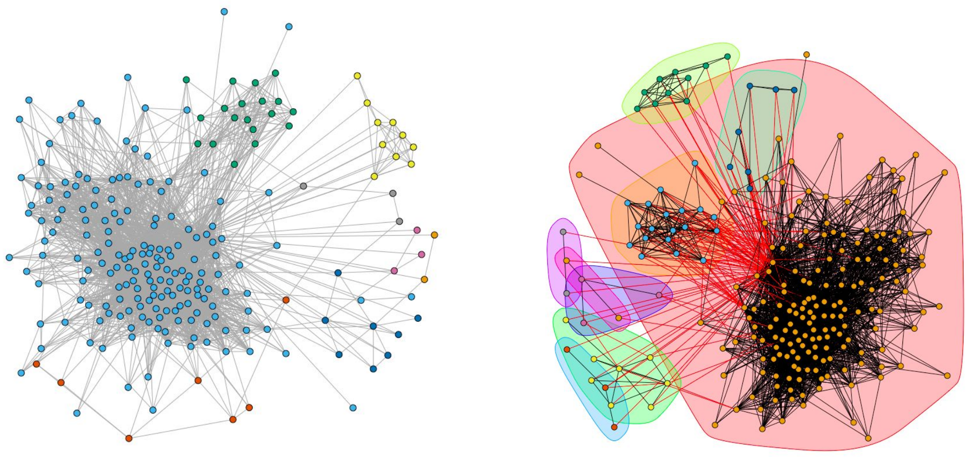


Figure 3.3 Community structure using infomap algorithm

Discussion

From Table 3.1, we can see that Fast-Greedy has the largest modularity value, which is also almost twice that of Infomap, three times that of Edge-Betweenness. This shows Fast-Greedy can detect and split the communities more reasonably. There are a lot of connection within one community and there are few connections between communities.

From Figure 3.1~3.3, we can see intuitively that Edge-Betweenness has many single node communities. Although Infomap does not have single node community, it has community size skew. In contract, Fast-Greedy presents a more reasonable community structure. One thing is that although Edge-Betweenness has larger modularity, it has a lot of single node communities. So to this degree, it may not be better than Infomap.

Question 4

Try removing the core node itself from its personal network and running the above community detection algorithms again. Are there any differences in the results?

Solution

Compared to Question 3, this question removes the core node from the core node's personal network. This may have a great impact on the personal network due to that all the other nodes in the personal network connect to the core node and if the core node is removed, all its edges will also be removed. This may effact the whole personal network structure and may let the personal community unconnected. The community figures of this question for USER with Id 3 are shown in Figure 4.1~4.3 for the three algorithms respectively. And the structural feature results are shown in Table 4.1.

Discussion

Compared to Question 3, the community structure for the personal network is quite different. Due to the lack of core node, all the core node edges are removed. So first, the network will become unconnected and there will exist some single node communities. Second, the modularity value will change. We can see from Table 4.1 that both Edge-Betweenness and Infomap modularity increase and Fast-Greedy decreases slightly. But Edge-Betweenness still has a lot of single node communities while the community structure from Fast-Greedy and Infomap seems more reasonable, which can also be seem from the modularity values.

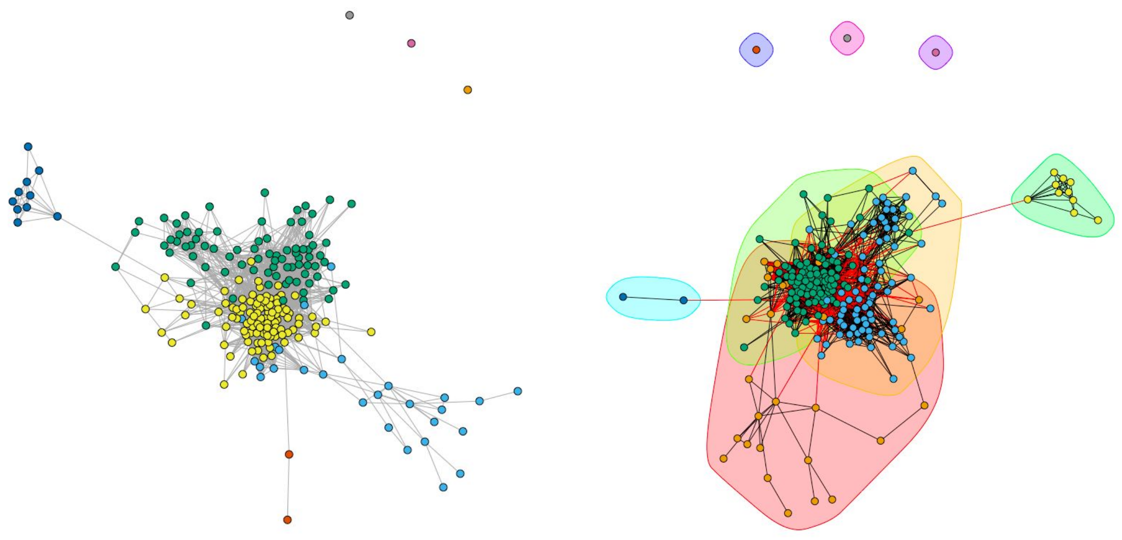


Figure 4.1 Community structrue without core node using Fast-Greedy

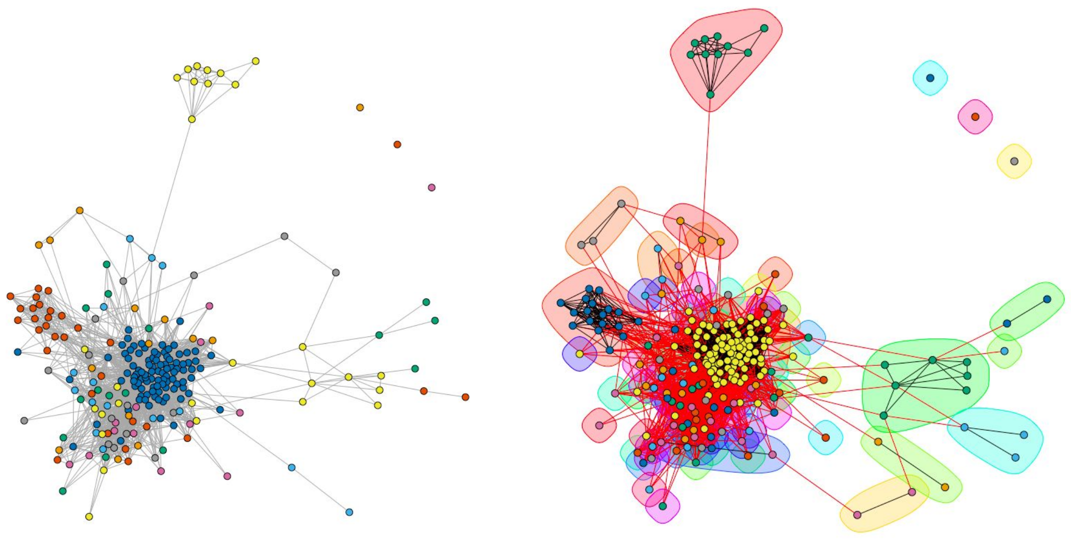


Figure 4.2 Community structure without core node using Edge-Betweenness

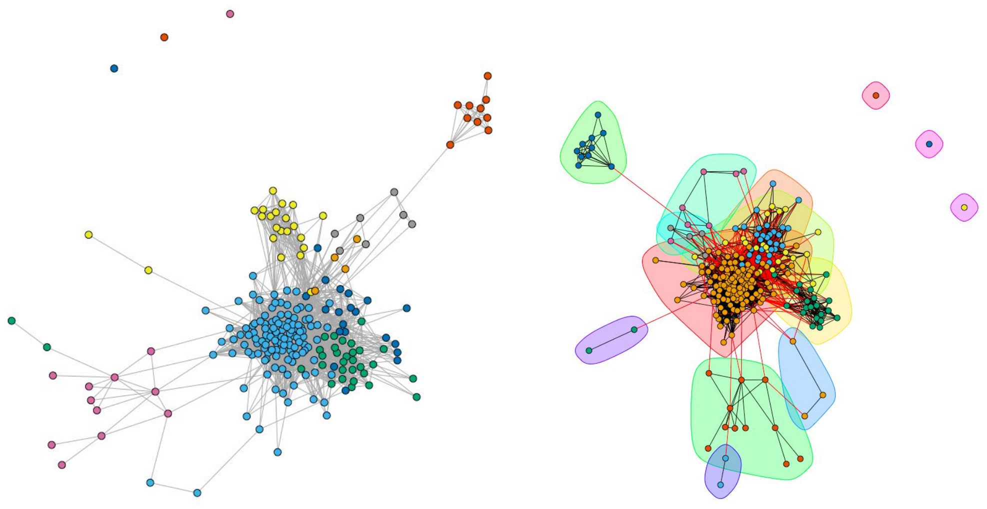


Figure 4.3 Community structure without core node using Infomap

Table 4.1 Structural features using three algorithms without core node

Algorithm	Fast-Greedy	Edge-Betweenness	Infomap
Modularity	0.2456918	0.1505663	0.2465785

Question 5

Find dispersion and embeddedness for all nodes in the personal network. Plot the distribution of embeddedness and dispersion over all personal networks created with core nodes from part 3. Plot 3 personal networks showing their community structure with colors and highlight the node with maximum dispersion in each network. Highlight the edges incident to this node as well. On each network, do the same thing for the node with maximum embeddedness and the node with maximum dispersion embeddedness.

Solution

Embeddedness is the number of mutual friends a node shares with the core node. High embeddedness implies that the two nodes have a lot of friends in common. This often happens when the two persons are in the same school or college. For example, most of my friends on Facebook with who I have a number of mutual friends, are from my IIT Guwahati, from where I did my undergrads.

On the other hand, dispersion is the sum of distances between every pair of the mutual friends a node shares with core node. This is a more formal way of saying that two persons know each other, but their mutual friends are not closely related.

Design and Assumptions

For each core node found in part 3, we calculate embeddedness and dispersion for each node in the respective personal network. Embeddedness is computed by taking intersection of the mutual friends a node u has with the core node v.

Dispersion requires a distance metric to compute the distance between two mutual friends of u and v. We considered two different metrics for this: 1) shortest path distance, i.e. the minimum number of hops between two nodes, and 2) metric given in [1]; if two nodes are not in direct contact with each and do not share any mutual friends other than u and v. We believe, the following experiments can help us get deeper insight into the pros and cons of each distance metric.

Results

Distribution of dispersion over all personal networks of the core nodes are shown in Figure 5.1 and 5.2 for shortest path distance and distance metric in [1], respectively. As expected, the dispersions calculated using shortest path distances are more evenly distributed. The distance metric in [1] yields dispersions that are extremely skewed towards 0. This is because, if two nodes have mutual friends other than u and v, it simply assigns a distance of 0. Authors in [1] have considered this as the best metric.

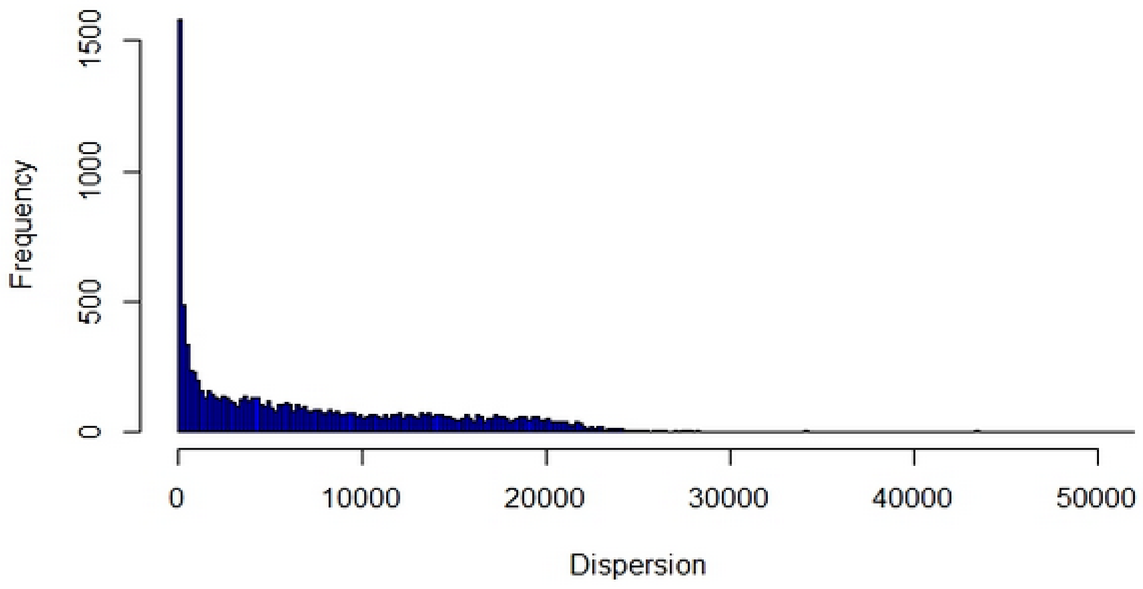


Figure 5.1 Distribution of dispersions, calculated using **shortest path** distance metric

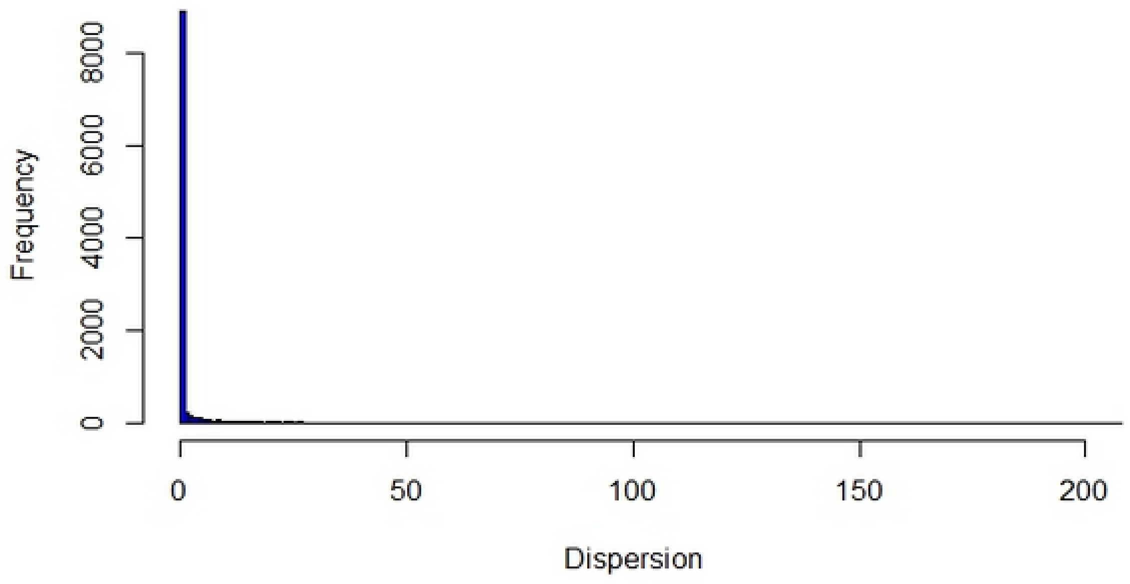


Figure 5.2 Distribution of dispersions, calculated using distance metric given in [1]

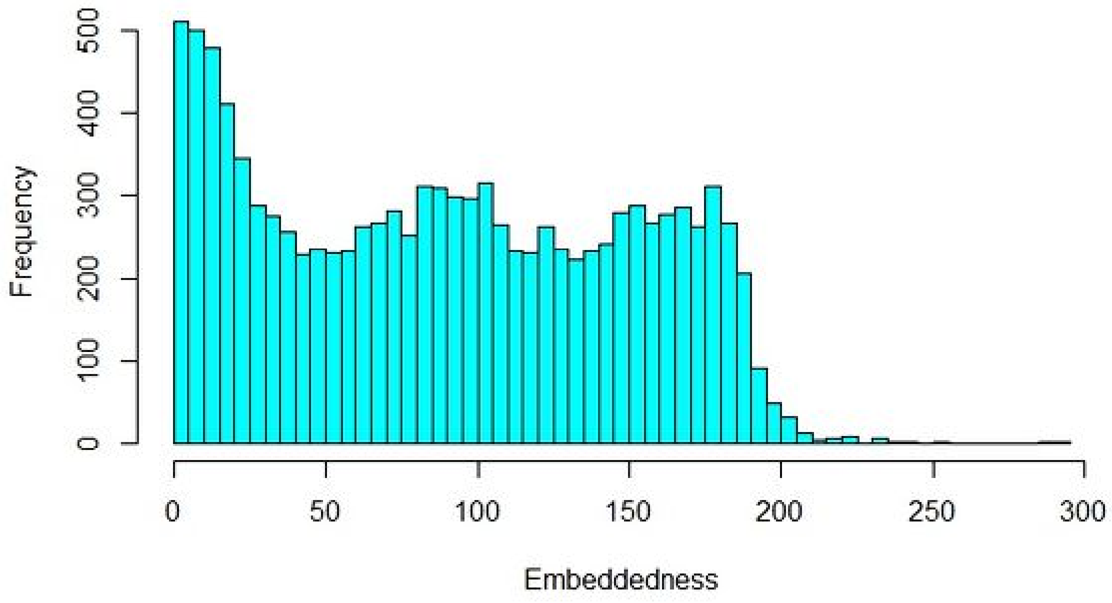


Figure 5.3 Distribution of embeddedness of over personal networks of all core nodes

Next we choose 3 core nodes 1, 4 and 12, and present the plots of three personal networks showing the core nodes and the nodes having maximum embeddedness, dispersion and dispersion/embeddedness ratio. Core node is shown as a large white node. The node having maximum embeddedness, dispersion and ratio are large nodes but of the same color as other nodes in the community. Please note that since we are considering two different distance metrics, we have two plots each of dispersion and ratio.

Core Node: 1

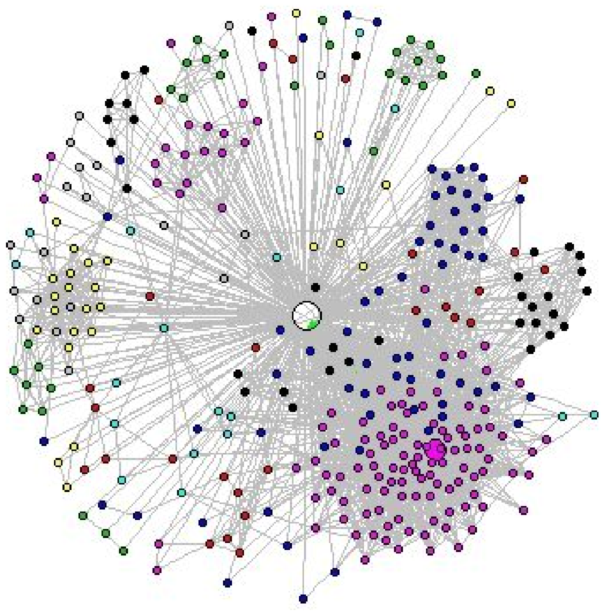


Figure 5.4 Personal network of core 1: maximum embeddedness node highlighted

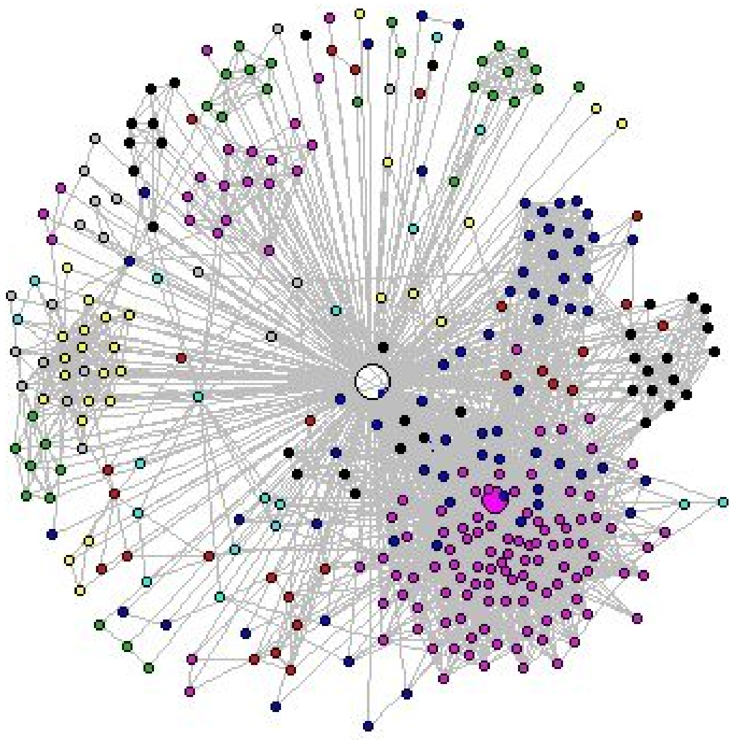


Figure 5.5 Personal network of core 1: max dispersion (calculate using [1]) node highlighted

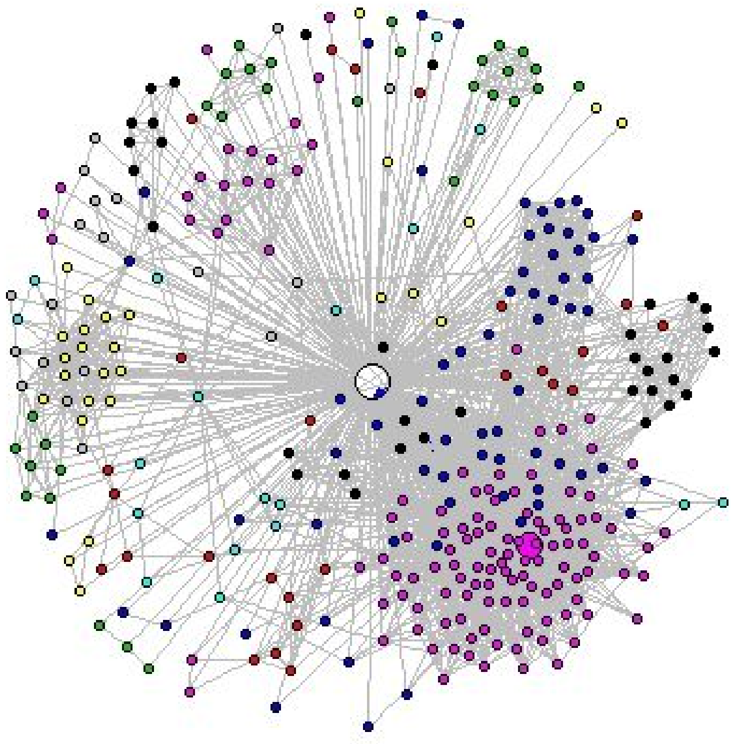


Figure 5.6 Personal network of core 1: max dispersion (using SP) node highlighted

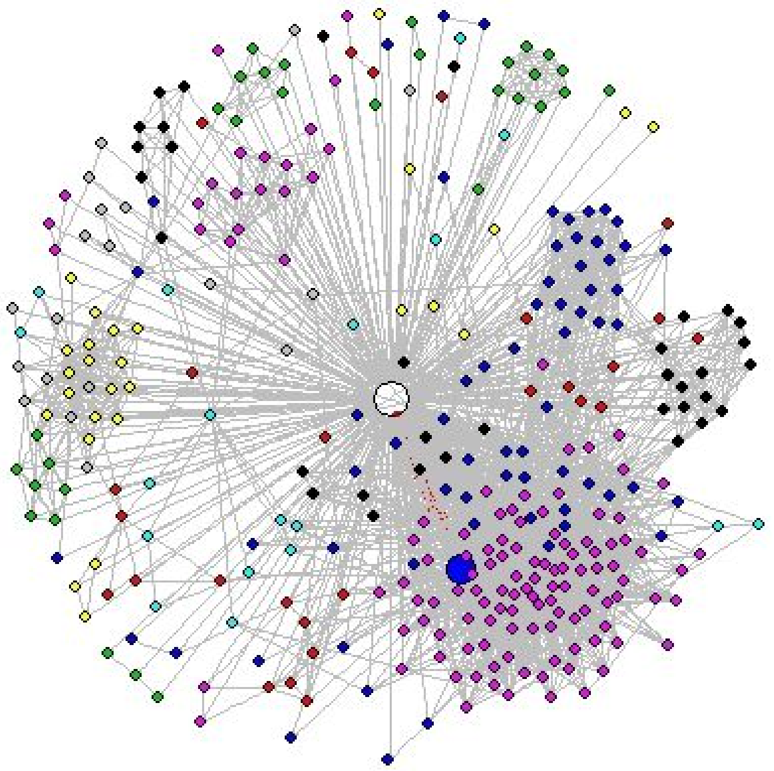


Figure 5.7 Personal network of core 1: max disper/embedded (using [1]) node highlighted

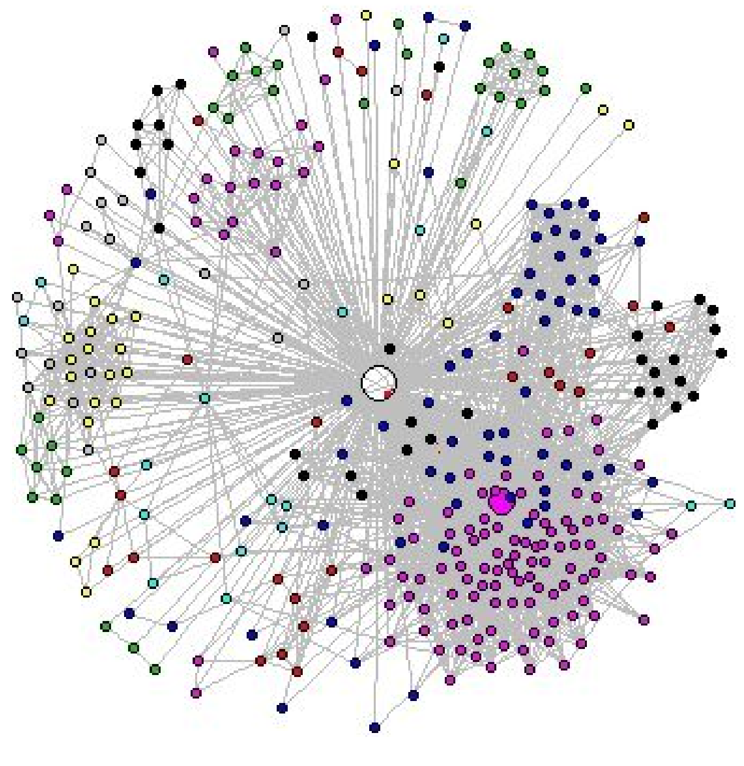


Figure 5.8 Personal network of core 1: max disper/embedded (using SP) node highlighted

Core Node: 4

Different distance metric need not always produce different results. Unlike core node 1, core node 4 dispersion values are robust to distance metric. This can be seen that both distance metrics, shortest path and [1], give same nodes as having the highest dispersion values. The nodes having highest value of dispersion/embeddedness ratio also remains the same.

Our objective of including this core node in the report is that we want to illuminate an interesting fact. In this personal network, the same node has the highest embeddedness, dispersion, and the dispersion/embeddedness ratio.

This imples that the two nodes have large number of mutual friends and most of those mutual friends do not know each other. This generally happens in the following scenario.

Both students, u and v, went to the same universities, U_A and U_B. Therefore they have a large number of mutual friends. But their friends in U_A do not know their friends in U_B. Therefore, the dispersion values are large. As dispersion values are calculated for every pair of mutual friends, this increases much faster than the number of mutual friends. More formally, I would say that the maximum possible value of dispersion can be ${}^{n}C_2$ for n mutual friends. This high value of dispersion outweighs the high value of embeddedness, yielding the high value of dispersion/embeddedness ratio.

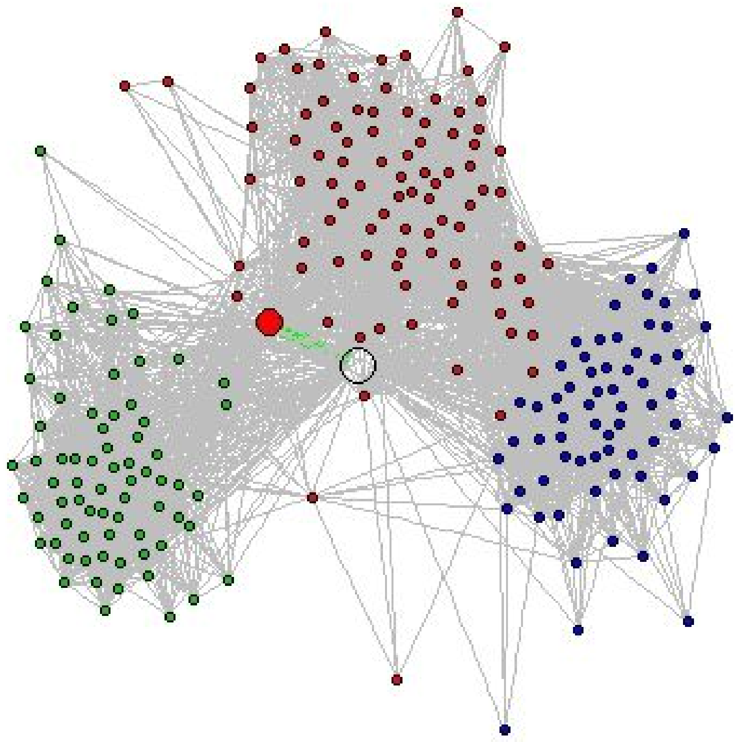


Figure 5.9 Personal network of core 4: maximum embeddedness node highlighted

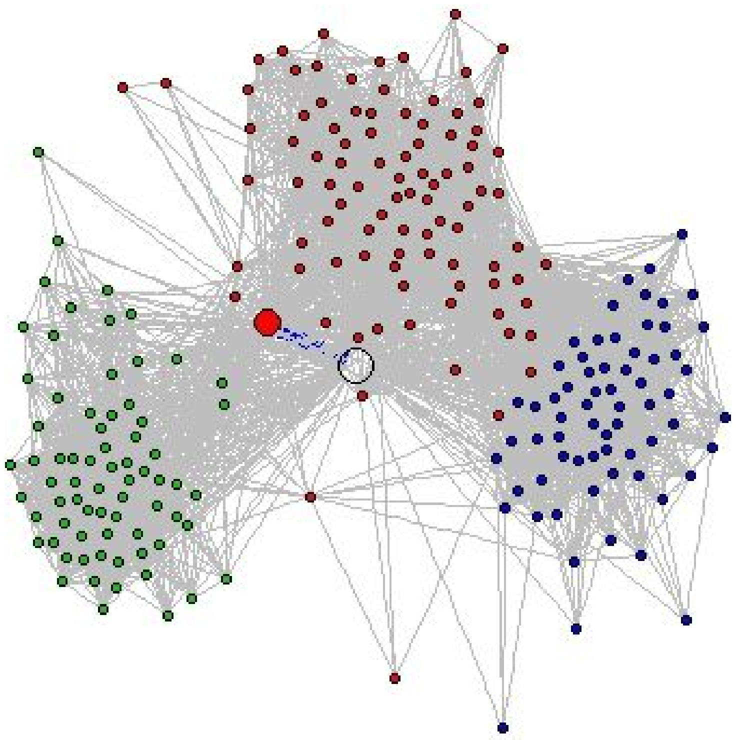


Figure 5.10 Personal network of core 4: max dispersion (using [1]) node highlighted

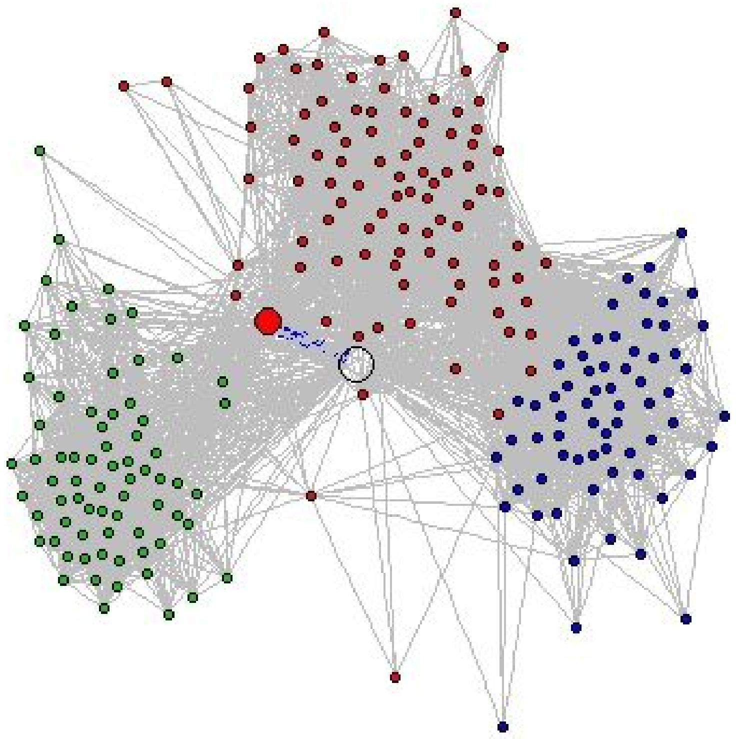


Figure 5.11 Personal network of core 4: max dispersion (using SP) node highlighted

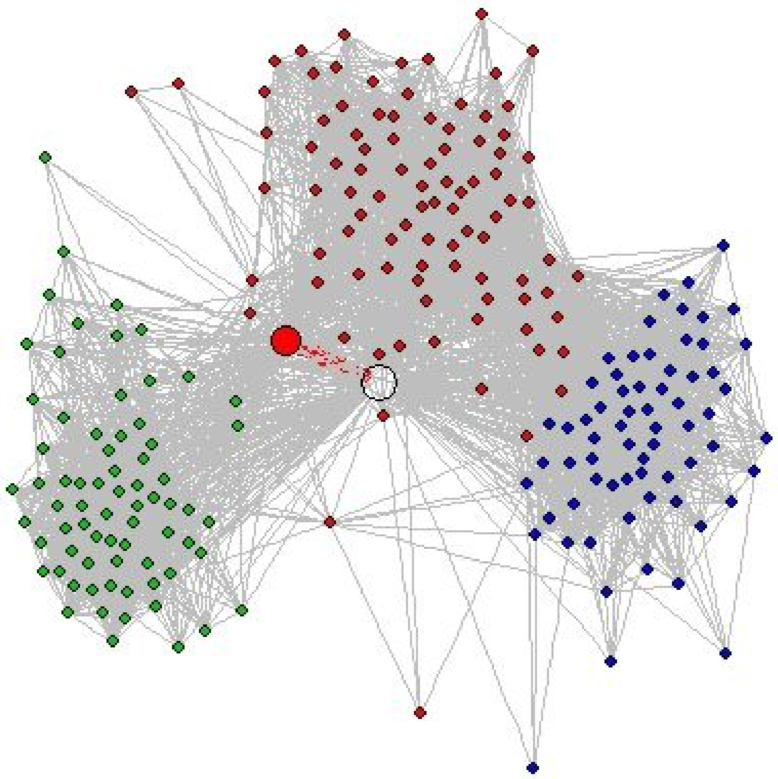


Figure 5.12 Personal network of core 4: max disper/embedded (using [1]) node highlighted

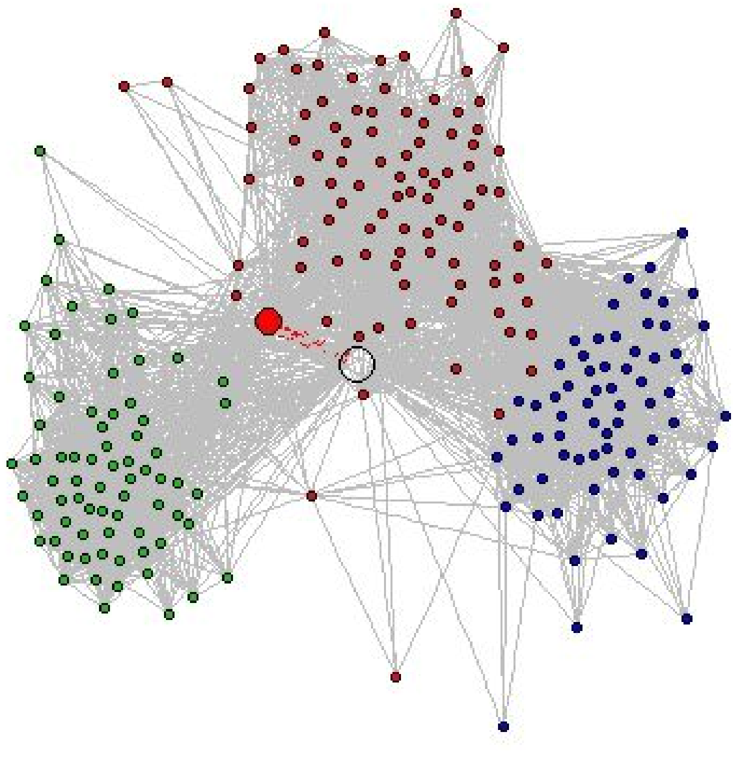


Figure 5.13 Personal network of core 4: max disper/embedded (using SP) node highlighted

Core Node 12

This is another intersting example in which we want to show how different distance metric can lead to very different results. Embeddedness values remain the same in the two case as distance metrics have nothing to do with embeddedness.

Dispersion values ae very different, and this can be easily seen that the nodes having maximum values of dispersions are different when using different distance metrics (Figure 5.15 and Figure 5.16). The two metrics give different results when there are a lot of mutual friends who directly know each other (not through any other friend). In this case, shortest path distance value between the mutual friends is 1. If these are added over all pairs of mutual friends, this gives a high value of dispersion. On the other hand, distance metric in [1] assigns a zero as the distance between mutual friends, and when added over all such pairs of mutual friends, the dispersion still remain zero. Therefore in this this, shortest path may mark the node as the one having highest value of dispersion, but [1] does not mark it as highest dispersion node; In fact, it considers that as the minimum possible value of dispersion.

These different distance metrics also mark different nodes as having maximum ratio. This is just a consequence of different distance metrics marking different nodes as having with maximum dispersion.

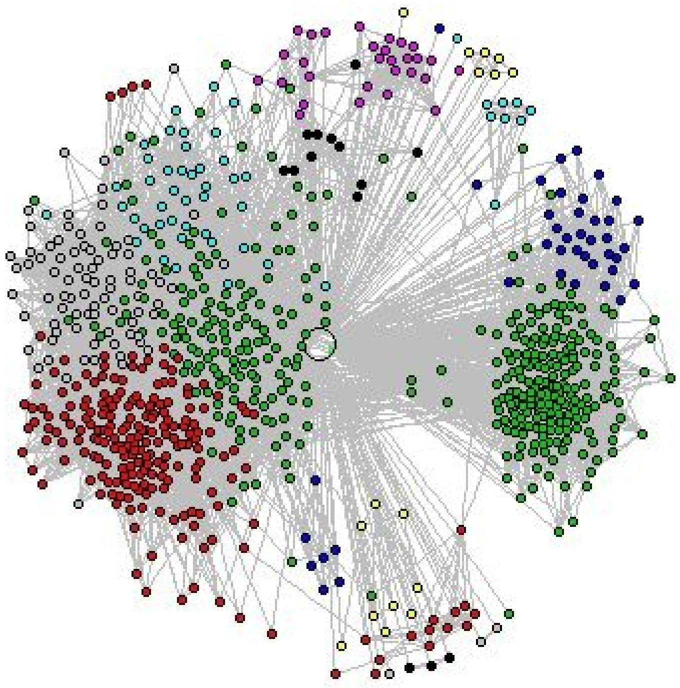


Figure 5.14 Personal network of core 12: maximum embeddedness node highlighted

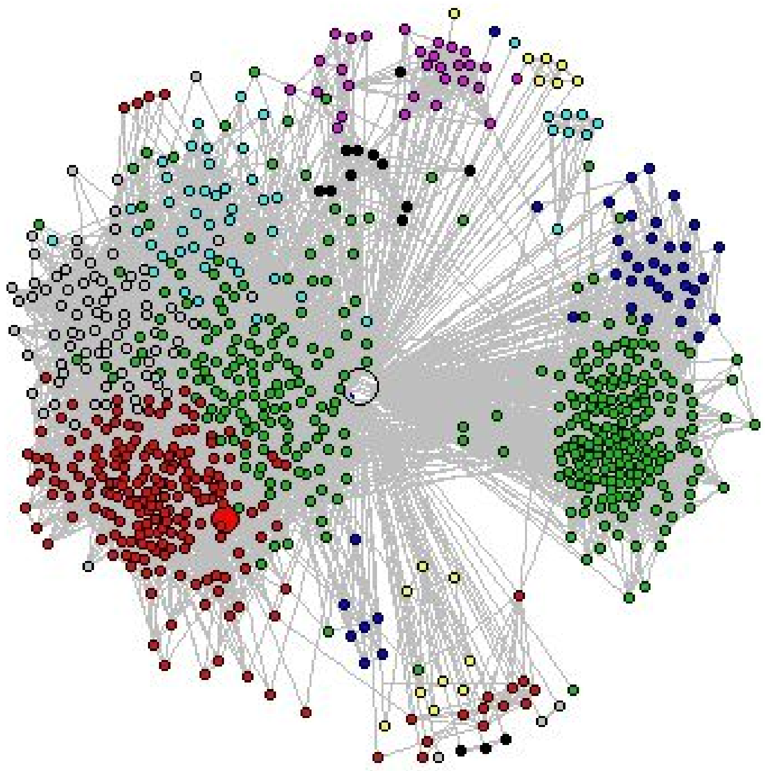


Figure 5.15 Personal network of core 12: maximum dispersion (using [1]) node highlighted

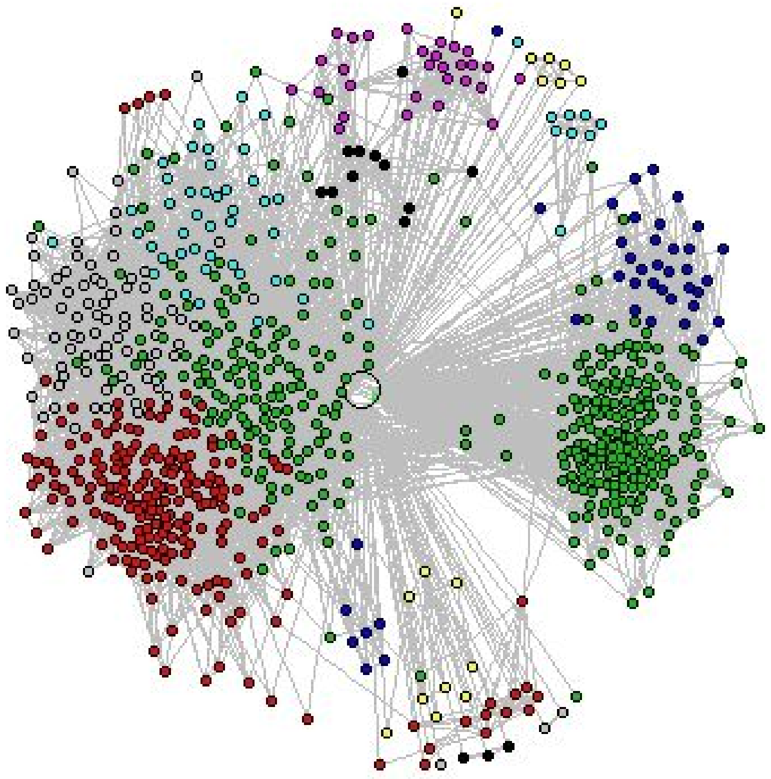


Figure 5.16 Personal network of core 12: maximum dispersion (using SP) node highlighted

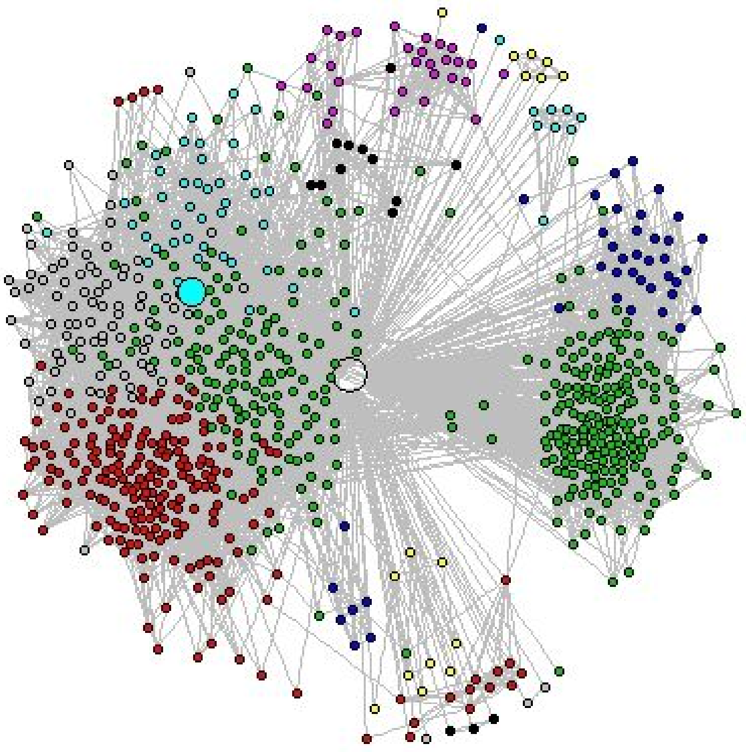


Figure 5.17 Personal network of core 4: maximum dispr/embed (using [1]) node highlighted

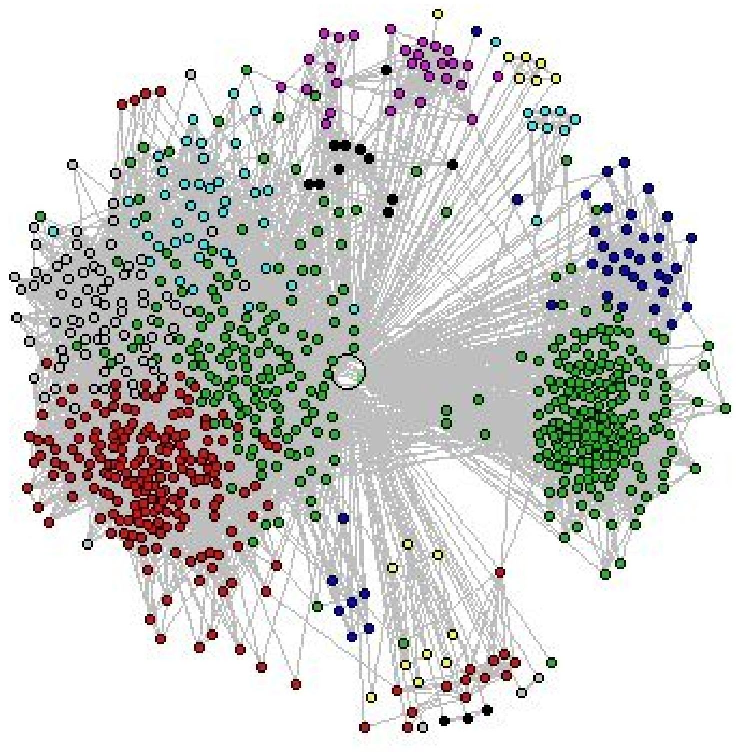


Figure 5.18 Personal network of core 12: maximum dispr/embed (using SP) node highlighted

Discussion

Embeddedness as I mentioned earlier is the number of mutual friends someone shares with the core node. This is not a good metric of romantic relationship as most often we tend to share more number of mutual friends with those who study with us in the same class or work in the same company as us. I tend to share maximum number of mutual friends with someone with who I did my undergrads.

Dispersion is the extent to which two people's mutual friends are not themselves well-connected. The sum of distances between the mutual friends is a good metric to quantify dispersion. This might be a decent metric of romantic relationship. However, people who have large number of mutual friends tend to have larger sum of distances.

Therefore to get a normalized metric of romantic relationship, we divide the the sum of distances i.e. dispersion by the number of mutual friends i.e. embeddedness.

<u>Romantic Relationships:</u>

During our experiments we found a few core nodes who most likely had romantic relationship with someone in their personal network. One of such core node personal network is shown in Figure 5.19. The two nodes are the bridging link between the two otherwise diconnected communities, and therefore are most likely in a romantic relationship.

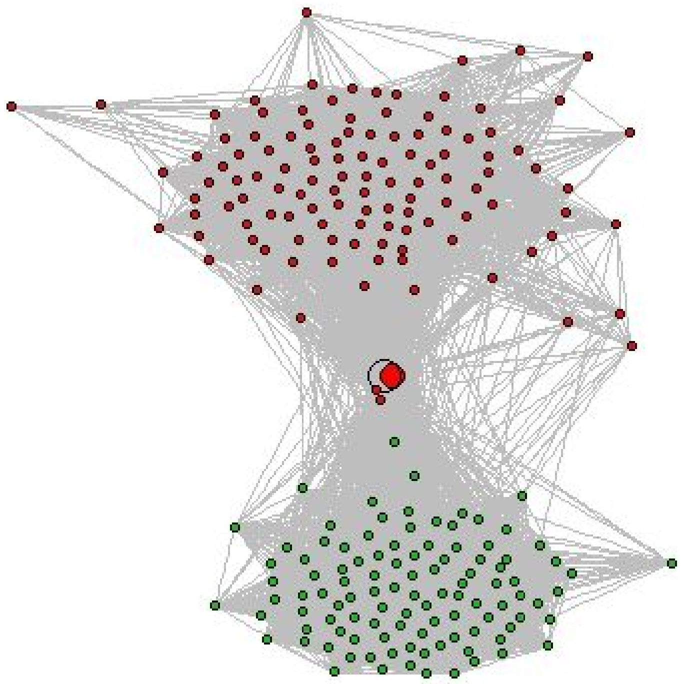


Figure 5.19. Personal network of core node 20: Core node and highlighted node bridges the otherwise disconnected communities.

Challenges

The major challenge was dealing with infinites in case of using shortest path as the distance metric. When there is no path between two nodes, the shortest distance between them is consdiered as infinity. Therefore the dispersion in that case in infinity. When there are multiple nodes with infinite dispersion in the personal network, it become challenging to decide which of them is a better candidate for maximum dispersion, for maximum dispersion/embeddedness ratio , and hence for romantic relationship. We circumvented this by summing only the finite distances when calculating the dispersion.

Another approach to deal with this would be to assign a high finite value to such distances. This “high finite value” can be one more than the diameter of the personal network. We found that this does yield different results. For example, below is the personal network of core node 1 showing the maximum ratio node, in which the above mentioned modified shortest path distance metric is used to calculate the dispersion. This can be seen that the node having maximum ratio is different from the one shown in Figure 5.8.

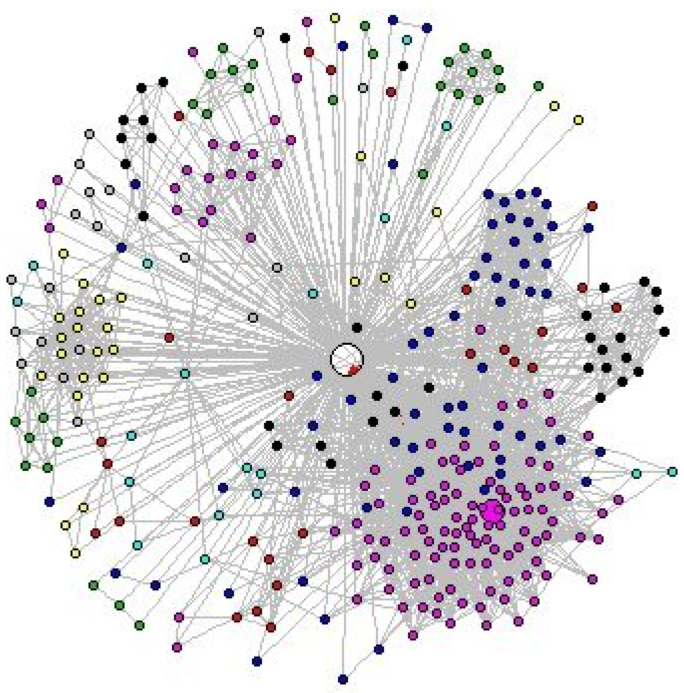


Figure 5.20 Personal network of core 1: max disper/embedded (using SP) node highlighted

Question 6

Run all personal networks and find two types of communities which are recurring across all of them, along with the features calculated. Try to show similarities among both communities in all networks.

Design and Analysis

For this question, the key is what structural features we should choose and how to use these features to determine a community type.

The structural features can be ***modularity***, ***cluster coefficient***, ***density***, ***average degree***, ***community size*** and so on. The ***modularity*** can measure how well the community structure is formed. If the modularity is large, then there will be many connections within each community and few connections between different communities. So if we explore the community structure of one community of a personal network, the modularity can show the structure properties of the explored community. For example, one company may have different departments. People in different departments may have fewer connections than in the same department. But all these people belong to a company (which can be seen as a community). So if the modularity of a community (we can first use community detection algorithm to analyze this community and then calculate the modularity) of a personal network is large, then this community may belongs to a "company" community type. The ***cluster coefficient*** can measure the degree to which nodes in a graph tend to cluster together. So this can also show the structure properties of a commuinty. For example, if a community type is familiy, then all the members in the family tend to know almost all the other members in the family. So the cluster coefficient will be large. The ***density*** is similar to cluster coefficient but may represent less about structure information. The ***average degree*** is kind of very similar to density statistically. The ***community size*** may vary among different communities. So absolute community size may be of little importance while relative community size may contain useful information for determining a community type.

Based on the above analysis, we choose to analyze ***modularity***, ***cluster coefficient***, ***density*** and ***relative community size*** (which equals to community_size/personal_network_size) and then choose the best two of them as the structural features. And the largest and smallest value of these features may represent two special community types, which we take into consideration for our analysis.

Solution

Due to the large number of nodes and the requirement of skipping personal networks with size smaller than 10, we decide to choose core nodes as our algorithm object. Based on the previous analysis, there are 40 core nodes with degree of 279.4 on average. For each core node personal network, we use Fast-Greedy algorithm (based on the analysis in Question 3 and Question 4) to explore their community structure, calculate the structrual feature values and store them. For cluster coefficient and density, we can evaluate the reasonableness of choosing these two features by comparing their max value community index in each core node network. If their indices are close or the same, this feature can help us identify a community type. This result is shown in Table 6.1. And the feature values are shown in Table 6.2. For other features, we use the standard variance to evaluate the resonableness of these features and because all these values' upbound is 1, we can compare them directly. The results are shown in Table 6.3. We can see that max_cluster_coefficient and max_relative_community_size have the smallest standard variance which are also much smaller than others. Combined with the results in Table 6.1 and 6.2, we finally decide to choose ***cluster_coefficient*** and ***relative_community_size*** as the structural features. And the community with the largest cluster coefficient and the community with the largest relative community size in each personal network represent the two types we want to find.

Table 6.1 Structrual features indexes

Core Node#	1	2	3	4	5	6	7	8	9	10
Community Index for Max Density	7	6	4	5	1	1	1	2	1	2
Community Index for Max Cluster Coefficient	7	6	4	5	1	1	1	2	1	2
Community Index for Min Density	1	1	1	2	4	3	2	1	3	1
Community Index for Min Cluster Coefficient	1	8	3	1	4	3	2	1	3	1
Core Node#	11	12	13	14	15	16	17	18	19	20
Community Index for Max Density	1	1	2	2	1	1	1	1	1	1
Community Index for Max Cluster Coefficient	1	1	2	2	2	1	1	1	2	1
Community Index for Min Density	2	3	1	1	2	3	3	3	3	2

Community Index for Min Cluster Coefficient	2	3	1	1	1	3	2	2	3	2
Core Node#	21	22	23	24	25	26	27	28	29	30
Community Index for Max Density	5	1	1	1	1	2	1	2	1	2
Community Index for Max Cluster Coefficient	1	1	1	1	1	2	1	2	1	2
Community Index for Min Density	3	2	3	2	2	1	2	1	3	1
Community Index for Min Cluster Coefficient	3	2	3	2	2	1	2	1	2	1
Core Node#	31	32	33	34	35	36	37	38	39	40
Community Index for Max Density	3	1	1	2	1	2	1	2	1	2
Community Index for Max Cluster Coefficient	3	1	1	2	1	2	1	2	1	2
Community Index for Min Density	1	2	2	1	2	1	2	1	2	1
Community Index for Min Cluster Coefficient	1	2	2	1	2	1	2	1	2	1

Table 6.2 Structrual features value

Core Node#	1	2	3	4	5	6	7	8	9	10
Max Density	0.87	1	1	1	0.59	0.50	0.63	0.62	0.75	0.66
Max Cluster Coefficient	0.94	1	1	1	0.78	0.71	0.75	0.73	0.77	0.76
Min Density	0.07	0.08	0.14	0.08	0.11	0.37	0.36	0.53	0.45	0.49
Min Cluster Coefficient	0.32	0	0.43	0.40	0.39	0.54	0.69	0.68	0.62	0.68
Max Relative Community Size	0.33	0.46	0.47	0.48	0.42	0.38	0.61	0.50	0.52	0.60
Min Relative Community Size	0.02	0.002	0.02	0.01	0.10	0.31	0.39	0.50	0.06	0.40
Max Modularity	0.43	0.52	0.38	0.48	0.33	0.18	0.18	0.11	0.10	0.11
Core Node#	11	12	13	14	15	16	17	18	19	20
Max Density	0.67	0.76	0.60	0.54	0.62	1	0.73	0.90	0.67	0.79
Max Cluster Coefficient	0.76	0.84	0.71	0.67	0.72	1	0.81	0.88	0.76	0.85
Min Density	0.44	0.60	0.50	0.47	0.58	0.49	0.52	0.56	0.52	0.57

Min Cluster Coefficient	0.69	0.70	0.66	0.46	0.71	0.66	0.67	0.70	0.66	0.73
Max Relative Community Size	0.61	0.49	0.59	0.52	0.57	0.51	0.56	0.60	0.43	0.53
Min Relative Community Size	0.39	0.12	0.41	0.04	0.43	0.02	0.05	0.02	0.28	0.47
Max Modularity	0.15	0.08	0.11	0.14	0.11	0.11	0.12	0.10	0.12	0.06
Core Node#	21	22	23	24	25	26	27	28	29	30
Max Density	0.75	0.84	0.83	0.93	0.77	0.86	0.79	0.91	0.91	0.83
Max Cluster Coefficient	0.91	0.89	0.75	0.93	0.83	0.89	0.86	0.92	0.91	0.87
Min Density	0.11	0.61	0.28	0.47	0.67	0.43	0.77	0.74	0.77	0.52
Min Cluster Coefficient	0.40	0.75	0.57	0.70	0.78	0.66	0.83	0.82	0.85	0.74
Max Relative Community Size	0.38	0.53	0.58	0.54	0.61	0.59	0.53	0.44	0.48	0.56
Min Relative Community Size	0.01	0.47	0.02	0.03	0.39	0.15	0.47	0.12	0.05	0.44
Max Modularity	0.46	0.07	0.30	0.12	0.07	0.29	0.03	0.04	0.03	0.10
Core Node#	31	32	33	34	35	36	37	38	39	40
Max Density	0.88	0.83	0.83	0.78	0.86	0.82	0.91	0.84	0.94	0.85
Max Cluster Coefficient	0.90	0.86	0.87	0.83	0.89	0.86	0.92	0.86	0.94	0.88
Min Density	0.74	0.71	0.65	0.59	0.77	0.67	0.79	0.74	0.67	0.59
Min Cluster Coefficient	0.82	0.81	0.80	0.83	0.85	0.80	0.84	0.84	0.85	0.76
Max Relative Community Size	0.49	0.61	0.54	0.52	0.46	0.56	0.50	0.52	0.47	0.56
Min Relative Community Size	0.10	0.39	0.46	0.48	0.10	0.44	0.07	0.48	0.06	0.44
Max Modularity	0.04	0.04	0.04	0.06	0.03	0.05	0.03	0.04	0.04	0.08

Table 6.3 Standard variance for different feature results

Feature	Max Density	Max Cluster Coefficient	Min Denstiy	Min Cluster Coefficient	Max Relative Communit y Size	Min Relative Communit y Size	Max Modularity
Standard Variance	0.13	0.08	0.21	0.18	0.07	0.19	0.14

Discussion

From the above results, we can see that :

Table 6.1 shows the closeness of feature cluster coefficient and feature density, which in turn shows the reasonableness of these two values. And the indexes for cluster coefficient and density are almost the same. Intuitively, large density or cluster coefficient usually shows that there are a lot of collections between the community members. Such community can be friends, classmates or some types where members of this community tend to know each other well. We select the community with the largest cluster coefficient as one type we desire to find, which shows the strongest collection community and this community is usually the same type across different networks.

Table 6.3 shows the standard variance for different feature maximum values. It can show whether this feature value is common across different networks. If across different networks one feature maximum value' standard variance is small, it shows it is almost a constant for different networks. As a result, if we find communities whose value of this feature is the maximum within networks they belong to, these communities can be of the same type. So we finally choose ***max_cluster_coefficient*** and ***max_relative_community_size*** as the structural feature.

Challenge and future work

This question is kind of like research. We have to explore which features to use for determining community type and how. The most difficult part lies on finding a effective structrual features. For this part, we take several features like modularity, cluster coefficient, density, community size into consideration. We first think from the point of reality, i.e, what features different communities in real work may have. Then we try to use different metrics to evaluate each features and then select the best two. For the how part, we think that the max or min values may represent some special type. So we just calculate the max and min value of the selected two structural features and compare the result. Finally, we choose max which is better than min.

Due to the limited time, our model is kind of simple and linear, just considering one single structural features. We think if we can combine different structural features and use non-linear model, maybe we can get a model with higher performance. We can even try to use machine learning model to train the parameters in our model.

Question 7

Run the same kind of analysis on Google+ dataset. Create personal network for users who have more than 2 circles, and extract the community structure of each personal network using both Walktrap and Infomap algorithms and show how communities overlap with the user's circles.

Solution

First, we analyze the network and extract all users who have more than 2 circles, which is the default number when you create and account. For each ego user, we detect the number of circle, which is the number of relationship tags of that user, and find that there are 57 users whose circles are larger than 2. Following table indicates five IDs of these users.

Table 7.1 Sample UID with more than 2 circles

UID	num of circles
102615863344410467759	4
101541879642294398860	5
100962871525684315897	3
103236949470535942612	14
107223200089245371832	24

Then we run both the Walktrap and Infomap algorithms to identify communities and see how these communities overlap with the user's circles. We define the intersection of nodes in one community and one circle of a user to be the overlap of that community and the circle, and let the intersection number divide by the node number in that circle to be the overlap percentage.

overlap percentage = (# of intersection of nodes) / (# of nodes in circle)

Since running through all 57 users are very time consuming, may take several days to complete. Thus we randomly pick two of them, which are marked pink in the above table, UID 102615863344410467759 and 100962871525684315897 to show the result.

First we analyze user 102615863344410467759, the overlap is shown in the following tables.

Table 7.2 overlap of user 102615863344410467759 using Infomap

	Circle 1	Circle 2	Circle 3	Circle 4
Community 1	100%	100%	100%	100%

Table 7.3 overlap of user 102615863344410467759 using Walktrap

	Circle 1	Circle 2	Circle 3	Circle 4
Community 1	42.00%	32.43%	39.13%	41.67%
Community 2	0	0	0	0
Community 3	18.00%	16.22%	17.39%	16.67%
Community 4	6.00%	8.11%	6.52%	6.25%
Community 5	32.00%	40.54%	34.78%	33.33%
Community 6	0	0	0	0
Community 7	2.00%	2.70%	2.17%	2.08%

From the table above, we can see that for this user, if we use Infomap, we will get only one community containning all the friends, thus the overlap with all circles are 100%, but if we use Walktrap, we will get 7 communities. For this user, Walktrap tend to split the network into more sub communities. Above half of the communities using Walktrap have some overlap with circles, some of them have very strong overlap with circles. Also, for this user, we can see that for each community, it has similar percentage of overlap with each circle, while the overlap percentage across different communities veries in a large range. We may conclude that every community has different size, which result in the different overlap with circles across different communities. And each community, no matter big or small, consists of members from each circle, almost evenly according to the circle size, which may explain the similar overlap in each community.

Also, we show result of another user with 3 circles, UID=100962871525684315897, using both Infomap and Walktrap algorithms.

Table 7.3 overlap of user 100962871525684315897 using Infomap

	Circle 1	Circle 2	Circle 3
Community 1	99.58%	98.29%	97.92%
Community 2	0.42%	0.85%	0

Community 3	0	0	2.08%
Community 4	0	0	0
Community 5	0	0	0
Community 6	0	0.85%	0
Community 7	0	0	0

Table 7.3 overlap of user 100962871525684315897 using Walktrap

	Circle 1	Circle 2	Circle 3
Community 1	5.46%	14.53%	25%
Community 2	0	0	0
Community 3	0	0	0
Community 4	94.54%	85.47%	75.00%

From these two tables, we can see that for this user, Infomap tends to come up with more communities than Walktrap, and about half of the communities have some overlap with circles, both Infomap and Walktrap, some of them have very strong overlap with circles. Similarly, each community shares similar overlap while it differs across different communities. For this user, from table 7.3, we can see that community 1 has more overlap with circle 3 and community 4 has more overlap with circle 1. This means that community 1 consists of members from all three circle, but circle 3 is dominant and community 4 consists of all members from all three circles while members from circle 1 are dominant. Also, compared with the previous user, overlaps are quite differ from user to user. Each user has different pattern of overlap, and overlap also varies for different algorithms used.

Challenge and future work

This problem is to compare community with circles, and the most difficult part is to define the overlap of the community and the circle. In ths solution, we define it as the number of intersection nodes devided by the number of nodes in the circle to come up with the analysis. However, we can also define it as the number of intersection nodes devided by the number of nodes in the community, and analyze it from the view of communities.

References

[1] "Romantic Partnerships and the Dispersion of Social Ties: A Network Analysis of Relationship Status on Facebook", Lars Backstrom, Jon Kleinberg.
http://arxiv.org/abs/1310.6753