ANALYZING COMMUNITY DETECTION ALGORITHMS

A Project Report

Presented to

Prof. Katerina Potika

Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Class

CS 185C

By

Ketki Kulkarni & Samanvitha Basole

December 2016

Abstract

Over the past decade, the rise of internet has brought people closer. The collection of relationships between people is growing fast due to technological advancement, social awareness and digital interaction. Social networking sites like Facebook, Twitter, Snapchat, and Instagram have built societies virtually. Often these real-world networks are represented as graphs in which people represent nodes and relationships between them represent edges. This way we can easily store and represent complex, rich information. But retrieving such information using existing methods is a very expensive task. Therefore, there is a need to find effective, quick methods to analyze social networks. One such aspect is a community detection. There are four methods for graph clustering and many algorithms defined for each of those methods. In this survey, we are going to explore five hierarchical clustering algorithms on different social networks which allowed us to evaluate and analyze these networks.

I. Problem Formulation

Community in a large social network is a set of nodes grouped together such that the nodes within communities are densely connected internally than that of nodes belonging to different communities. Communities help us to find clusters of nodes which have common properties and evaluate relationships between them. For example, in a social network like Facebook people represent nodes, interactions among them represent edges and the communities are groups of people who follow same football club or support a same presidential nominee.

The study of community detection is a standard problem in analyzing large and complex social networks. Many researchers have concentrated in solving this problem. Some of the methods are graph partitioning, hierarchical clustering, partitional clustering and spectral clustering. Identifying such communities allows us to evaluate individual objects, interaction between them and predict missing information. Therefore, analyzing social network data in real-world networks and detecting communities have become a very important problem in various areas.

II. History and Background

As already mentioned, there are four methods of graph partitioning which are graph partitioning, hierarchical clustering, partitional clustering and spectral clustering. The graph partitioning method divides the graph in a set of vertices of predefined size by reducing the no. of edges between these sets. The hierarchical clustering method uses two approaches. One is top-down or divisive in which large network graph is iteratively broken down into different chunks based on centrality measures. Other approach is bottom-up or agglomerative in which vertices are combined into community iteratively so that at the end large communities are formed. The partitional clustering divides vertices into predefined number of clusters to maximize/minimize given cost function. And the spectral clustering uses eigenvectors of matrices to cluster points into one set.

III. Algorithms

In this survey, we have analyzed 5 different algorithms which comes under either an agglomerative hierarchical clustering or divisive hierarchical clustering. These algorithms are,

1. Girvan-Newman Algorithm

2. Louvain Algorithm
3. Label Propagation Algorithm
4. Random Walk Algorithm
5. Fast Greedy Algorithm

IV. Tools

1. Python
2. iGraph package
3. PyCharm/Canopy
4. GitHub

V. Source Code

The source code for this project, screenshots, final report, project proposal, and dataset files are available on GitHub: https://github.com/samanvithab/SocialNetworkAnalysis

VI. Datasets

1. Zachary's Karate Club
2. Albert-Barabasi Model
3. Erdos-Renyi Model
4. Facebook dataset

VII. Process

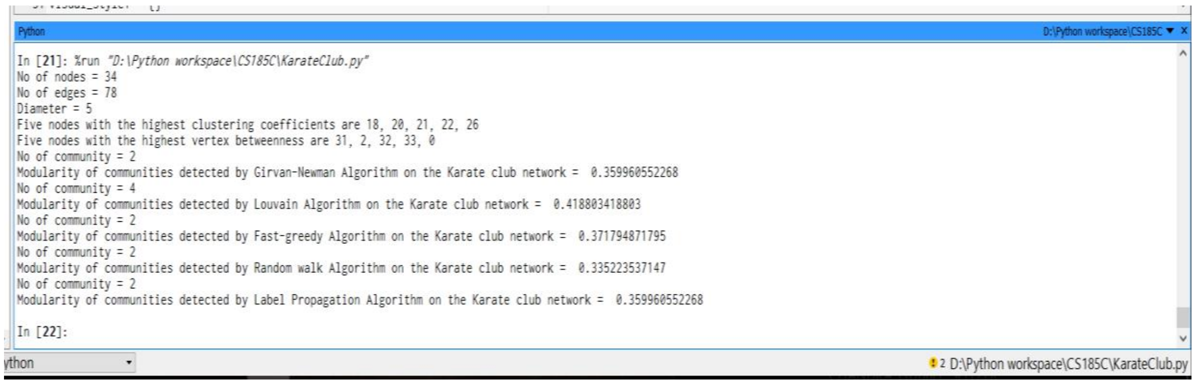
For the Karate Club, we started by passing in the GraphML file as an argument to the igraph.read() function. Then, we specified the visual style to make the graph more interesting. We plotted the original to compare the algorithms with the result. To do so, we had to call plot() on the graph instance created by the igraph.read() method. We called different algorithms in iGraph. For example, to create a graph using Girvan-Newman algorithm, we called the community_edge_betweenness() method. To plot the graph using the Louvain Method, we called community_multilevel() on the dataset. To use the Fast Greedy approach, we called the community_fastgreedy() method. To call the Random Walks algorithm, we called the community_walktrap() function. To use the Label Propagation approach, we called the community_label_propagation() method.

We repeated this process for the rest of the datasets. In case of Facebook dataset, we didn't apply fast-greedy algorithm as this algorithms does not work on the network with multiple edges.

VIII. Result

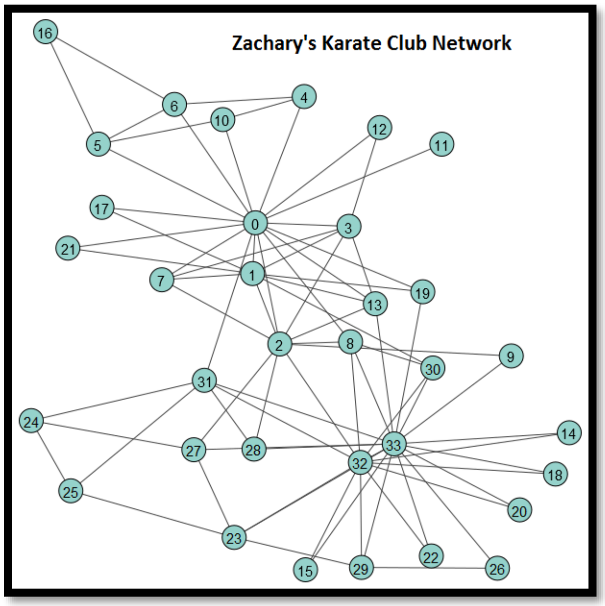
I. Zachary's Karate Club

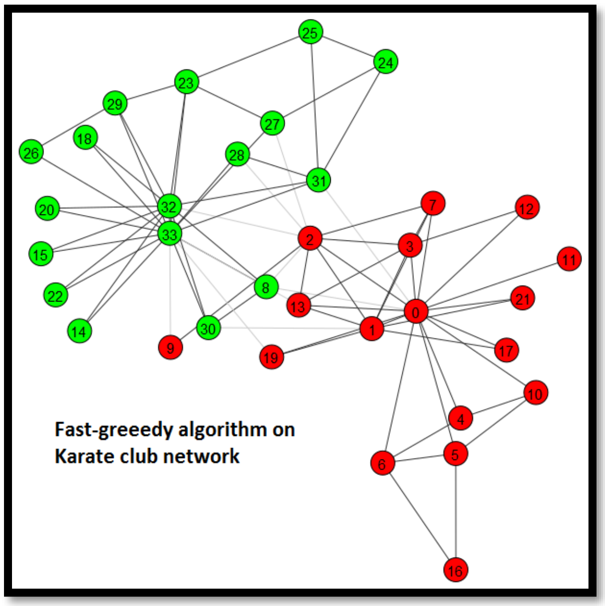
No. of nodes = 34
No. of edges = 78
No. of connected components = 1
Five nodes with the highest clustering coefficients = 18, 20, 21, 22, 26
Five nodes with the highest vertex betweenness = 0, 2, 31, 32, 33

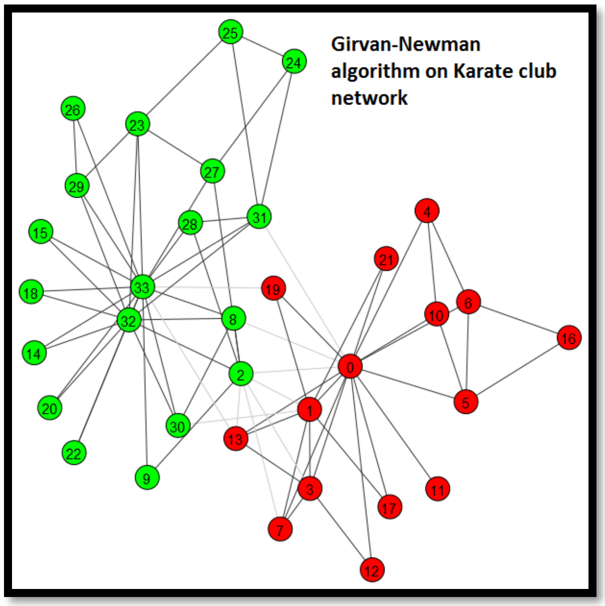


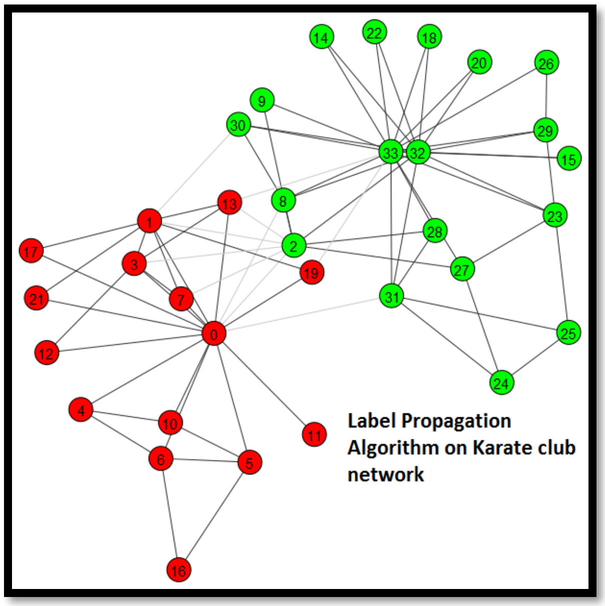
```
In [21]: %run "D:\Python workspace\CS185C\KarateClub.py"
No of nodes = 34
No of edges = 78
Diameter = 5
Five nodes with the highest clustering coefficients are 18, 20, 21, 22, 26
Five nodes with the highest vertex betweenness are 31, 2, 32, 33, 0
No of community = 2
Modularity of communities detected by Girvan-Newman Algorithm on the Karate club network =  0.359960552268
No of community = 4
Modularity of communities detected by Louvain Algorithm on the Karate club network =  0.418803418803
No of community = 2
Modularity of communities detected by Fast-greedy Algorithm on the Karate club network =  0.371794871795
No of community = 2
Modularity of communities detected by Random walk Algorithm on the Karate club network =  0.335223537147
No of community = 2
Modularity of communities detected by Label Propagation Algorithm on the Karate club network =  0.359960552268

In [22]:
```

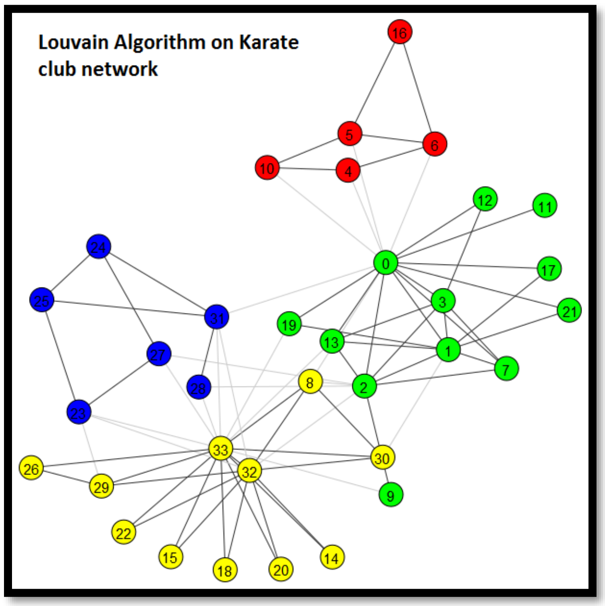
Python
D:\Python workspace\CS185C
ython
2 D:\Python workspace\CS185C\KarateClub.py

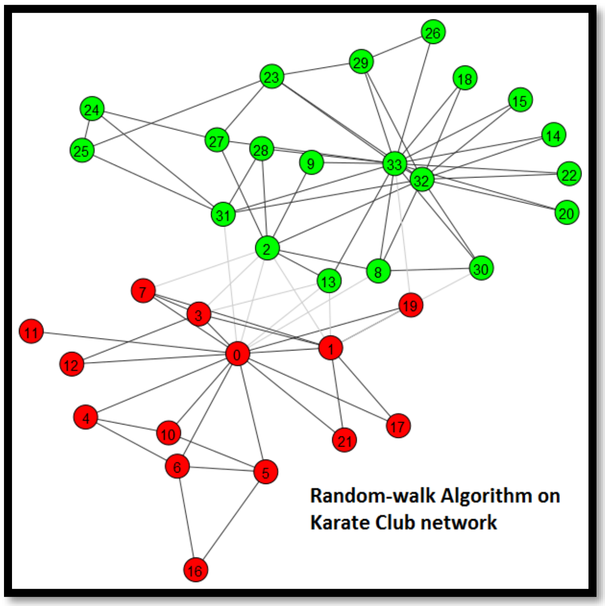

Zachary's Karate Club Network





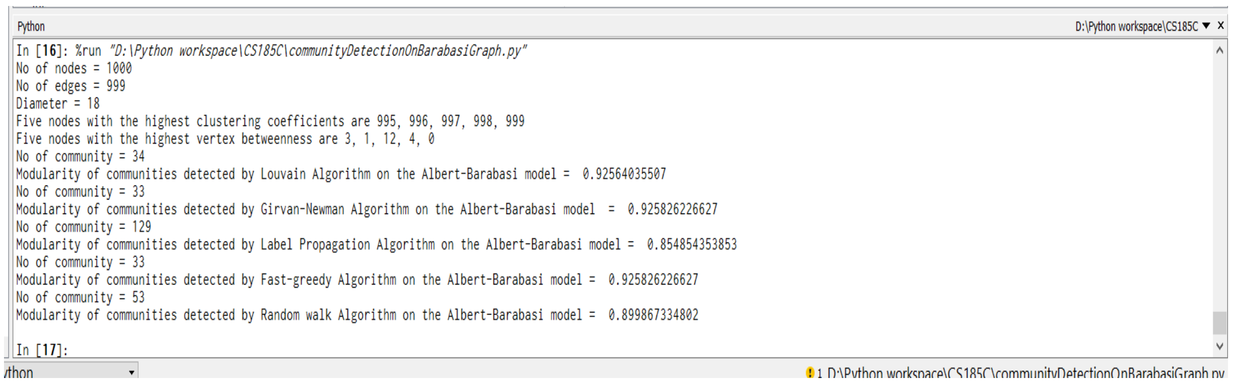






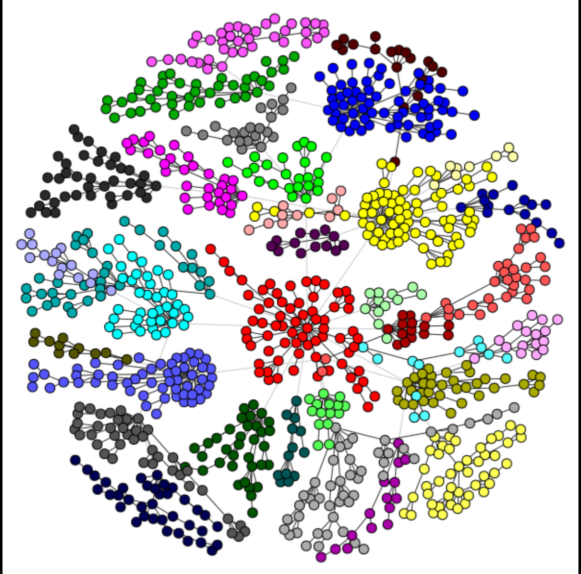
II. Albert-Barabasi Model

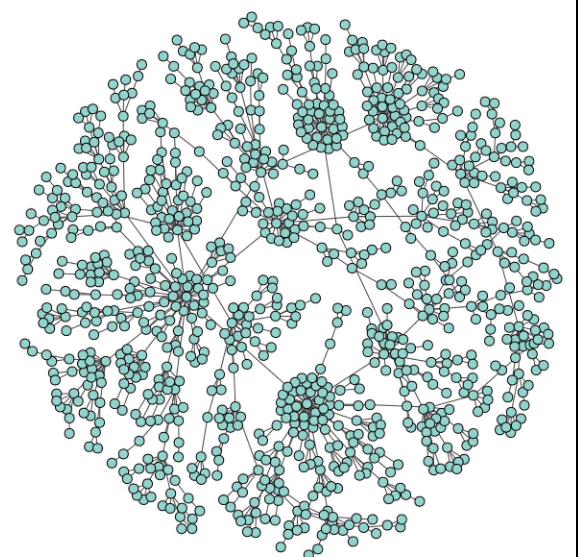
No. of nodes = 1000
No. of edges = 999
No. of connected components = 1
Five nodes with the highest clustering coefficients = 995, 996, 997, 998, 999
Five nodes with the highest vertex betweenness = 0, 1, 3, 4, 12

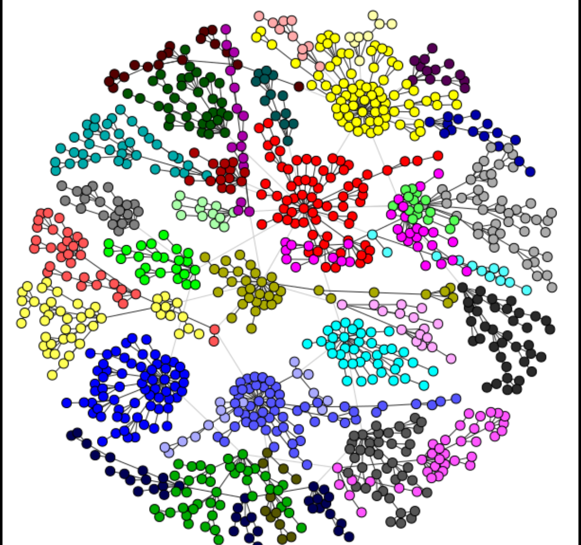


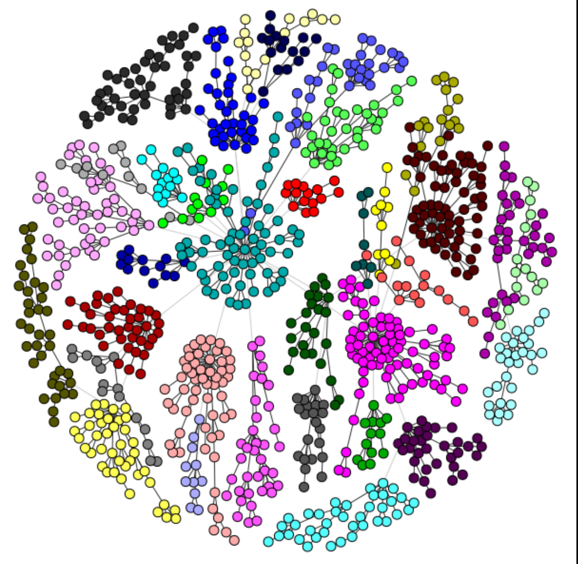
```
In [16]: %run "D:\Python workspace\CS185C\communityDetectionOnBarabasiGraph.py"
No of nodes = 1000
No of edges = 999
Diameter = 18
Five nodes with the highest clustering coefficients are 995, 996, 997, 998, 999
Five nodes with the highest vertex betweenness are 3, 1, 12, 4, 0
No of community = 34
Modularity of communities detected by Louvain Algorithm on the Albert-Barabasi model =  0.92564035507
No of community = 33
Modularity of communities detected by Girvan-Newman Algorithm on the Albert-Barabasi model  =  0.925826226627
No of community = 129
Modularity of communities detected by Label Propagation Algorithm on the Albert-Barabasi model =  0.854854353853
No of community = 33
Modularity of communities detected by Fast-greedy Algorithm on the Albert-Barabasi model =  0.925826226627
No of community = 53
Modularity of communities detected by Random walk Algorithm on the Albert-Barabasi model =  0.899867334802

In [17]:
```

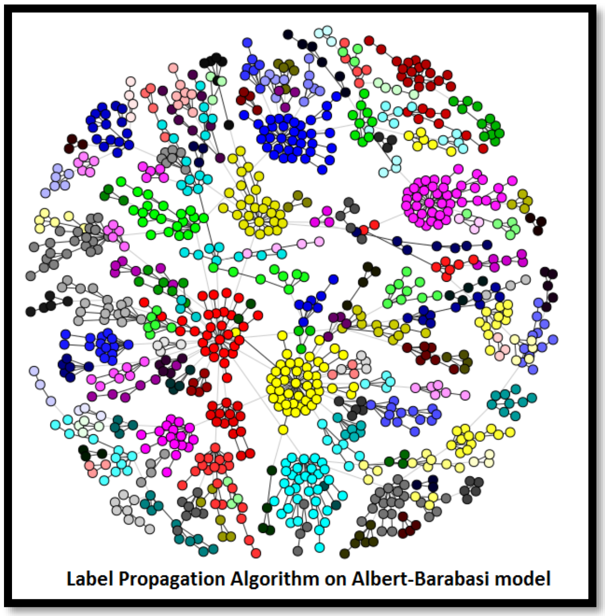
Python
D:\Python workspace\CS185C
ython
1 D:\Python workspace\CS185C\communityDetectionOnBarabasiGraph.py

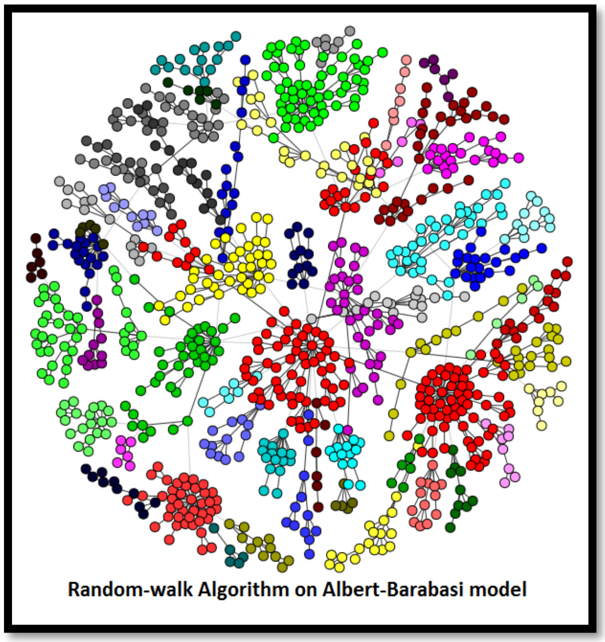

Fast-greedy algorithm on Albert-Barabasi Model


Albert-Barabasi Model


Girvan-Newman algorithm on Albert-Barabasi model


Louvain Algorithm on Albert-Barabasi Model


Label Propagation Algorithm on Albert-Barabasi model


Random-walk Algorithm on Albert-Barabasi model

III. Erdos-Renyi Model

No. of nodes = 300

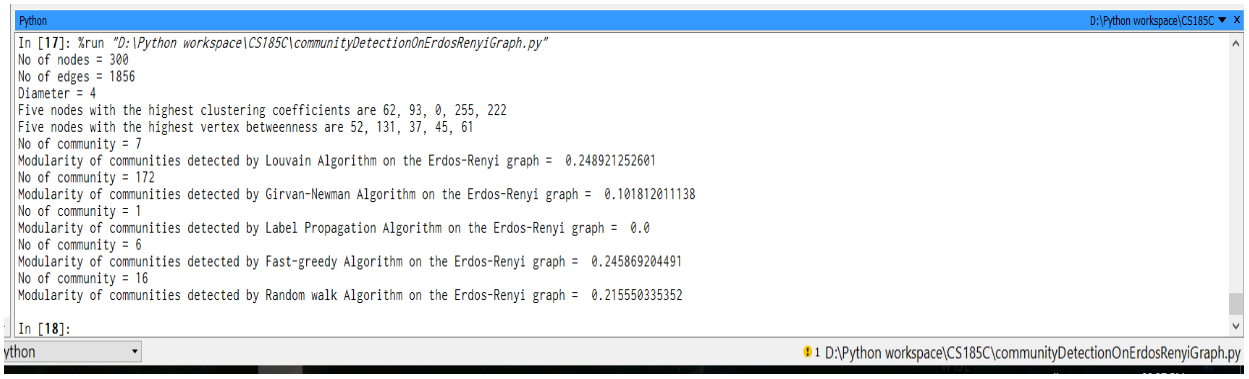
No. of edges = 1856

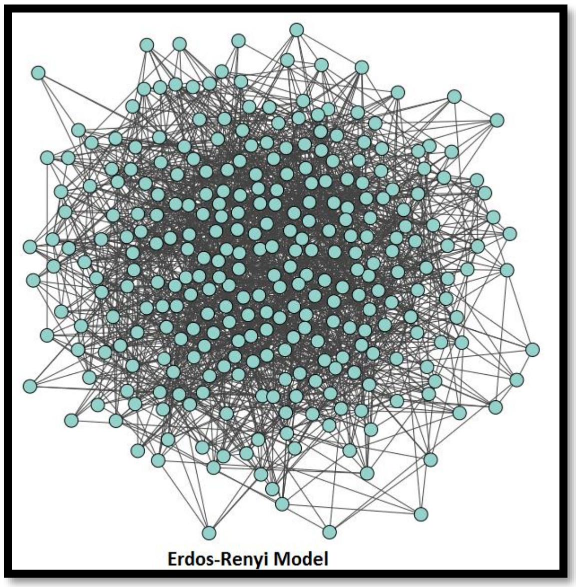
Probability of edges = 0.04

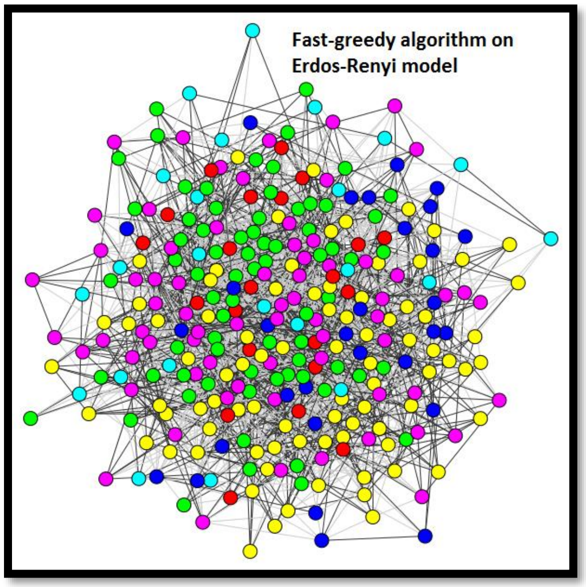
No. of connected components = 1

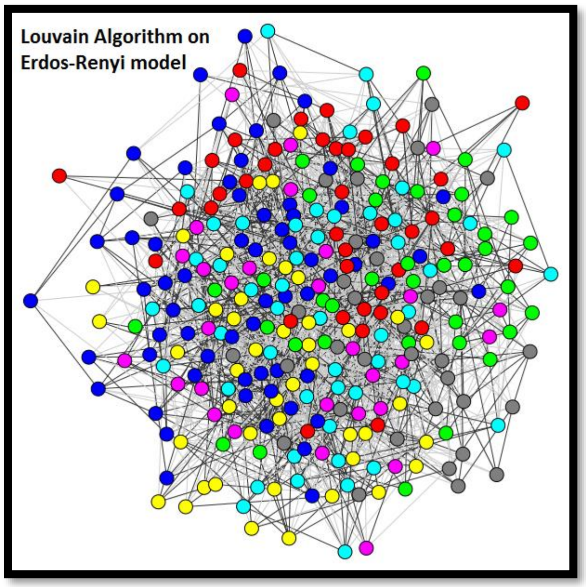
Five nodes with the highest clustering coefficients = 62, 93, 0, 255, 222

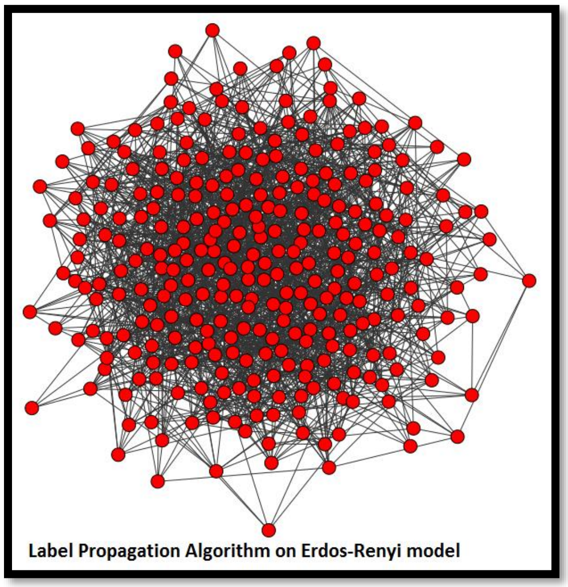
Five nodes with the highest vertex betweenness = 52, 131, 37, 45, 61

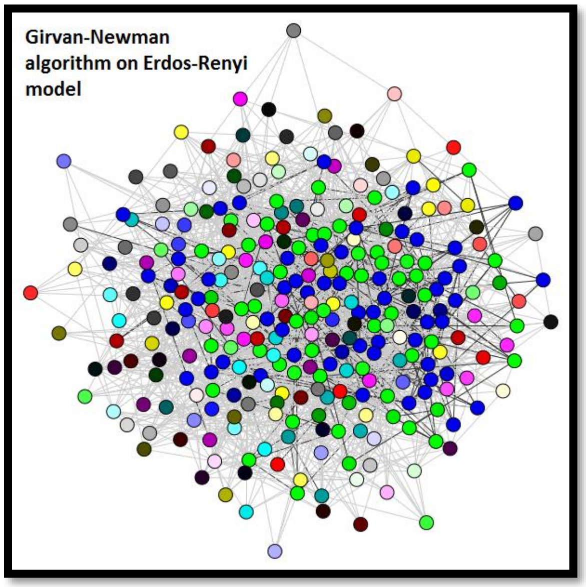

```
In [17]: %run "D:\Python workspace\CS185C\communityDetectionOnErdosRenyiGraph.py"
No of nodes = 300
No of edges = 1856
Diameter = 4
Five nodes with the highest clustering coefficients are 62, 93, 0, 255, 222
Five nodes with the highest vertex betweenness are 52, 131, 37, 45, 61
No of community = 7
Modularity of communities detected by Louvain Algorithm on the Erdos-Renyi graph =  0.248921252601
No of community = 172
Modularity of communities detected by Girvan-Newman Algorithm on the Erdos-Renyi graph =  0.101812011138
No of community = 1
Modularity of communities detected by Label Propagation Algorithm on the Erdos-Renyi graph =  0.0
No of community = 6
Modularity of communities detected by Fast-greedy Algorithm on the Erdos-Renyi graph =  0.245869204491
No of community = 16
Modularity of communities detected by Random walk Algorithm on the Erdos-Renyi graph =  0.215550335352

In [18]:
```
Python
D:\Python workspace\CS185C
ython
1 D:\Python workspace\CS185C\communityDetectionOnErdosRenyiGraph.py

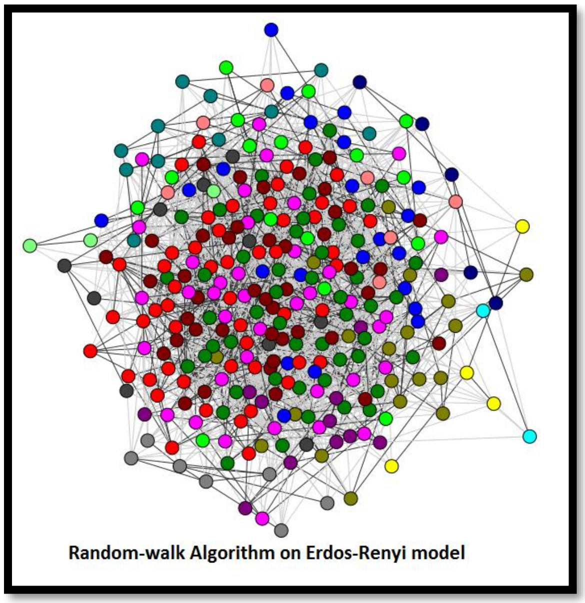

Erdos-Renyi Model


Fast-greedy algorithm on
Erdos-Renyi model


Louvain Algorithm on
Erdos-Renyi model


Label Propagation Algorithm on Erdos-Renyi model





IV. Facebook Network

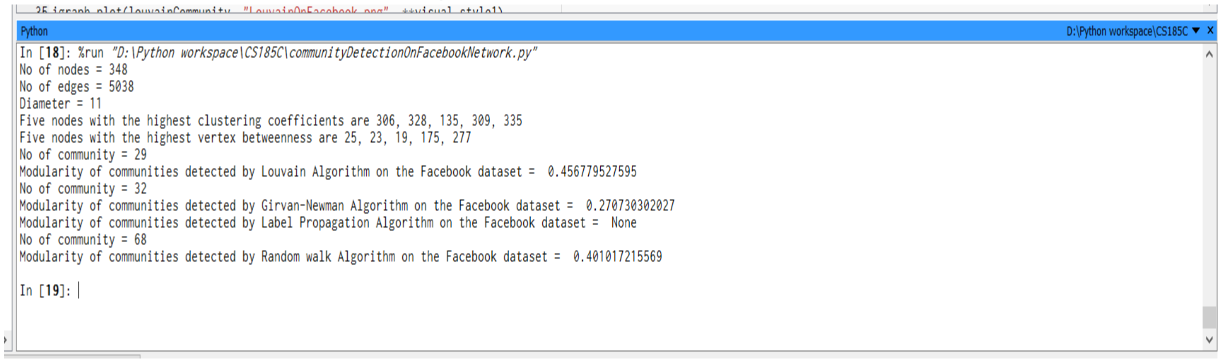
No. of nodes = 348

No. of edges = 5038

No. of connected components = 20

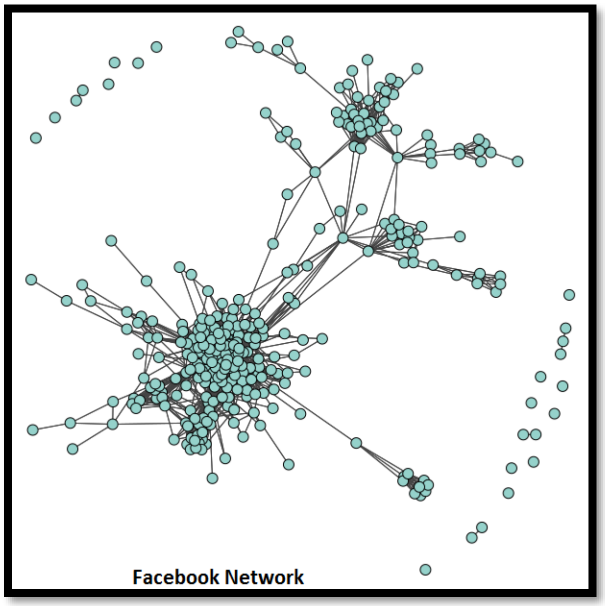
Five nodes with the highest clustering coefficients = 306, 328, 135, 309, 335

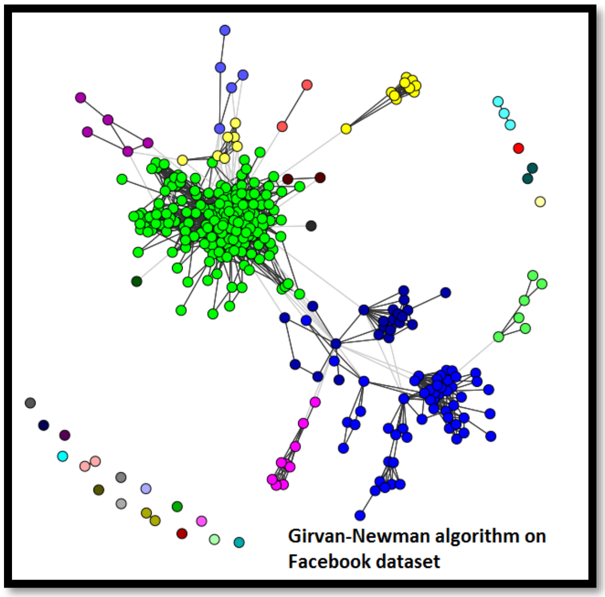
Five nodes with the highest vertex betweenness = 25, 23, 19, 175, 277

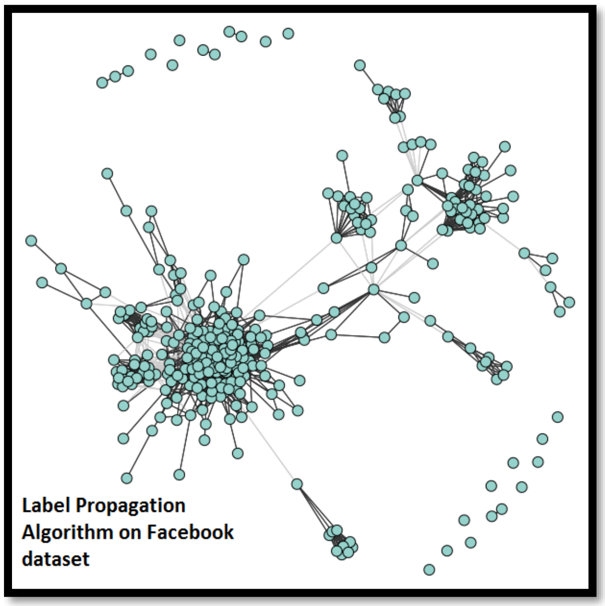


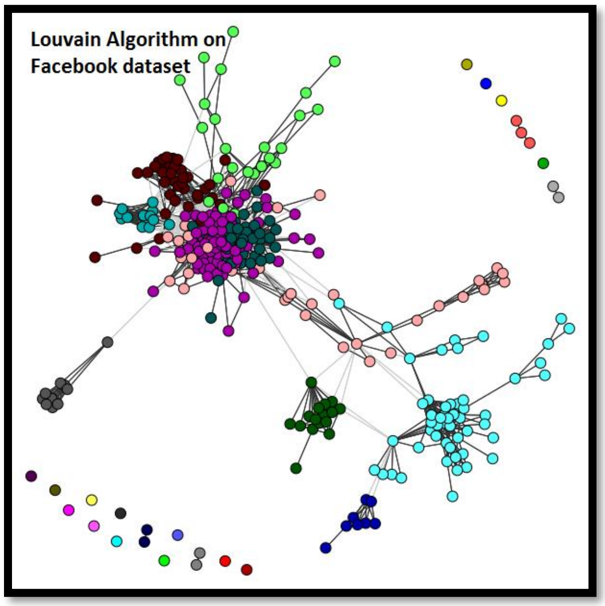
```
25 igraph.plot(louvainCommunity, "LouvainOnFacebook.png", **visual_style1)
```

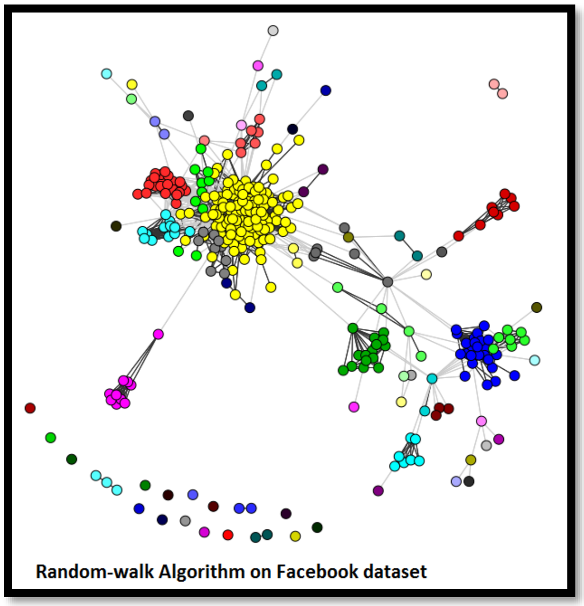
```
In [18]: %run "D:\Python workspace\CS185C\communityDetectionOnFacebookNetwork.py"
No of nodes = 348
No of edges = 5038
Diameter = 11
Five nodes with the highest clustering coefficients are 306, 328, 135, 309, 335
Five nodes with the highest vertex betweenness are 25, 23, 19, 175, 277
No of community = 29
Modularity of communities detected by Louvain Algorithm on the Facebook dataset =  0.456779527595
No of community = 32
Modularity of communities detected by Girvan-Newman Algorithm on the Facebook dataset =  0.270730302027
Modularity of communities detected by Label Propagation Algorithm on the Facebook dataset =  None
No of community = 68
Modularity of communities detected by Random walk Algorithm on the Facebook dataset =  0.401017215569

In [19]: 
```


Facebook Network


Girvan-Newman algorithm on Facebook dataset


Label Propagation Algorithm on Facebook dataset


Louvain Algorithm on Facebook dataset



IX. Challenges

During this project, we ran into some errors which took a while to fix. We also had some problems that we had a hard time in finding the solutions to. For example, we were not able to read a .txt file through iGraph. We solved this problem by searching for dataset files in other formats (GraphML and edges files). Another problem we had was that some of the functions were inputting a dendrogram object instead of the graph, but we could learn about the dendrogram object and figure out a way to convert the dendrogram object to a graph. We also spent some time figuring out how to plot the graphs using various layouts and visual elements such as node size, label size, node color, etc. We could successfully display colorful and neat graphs after reading the documentation and experimenting with different options.

X. Conclusion

With advancement of parallel computing, storage, and super-computing, use of these methods on large social networks is becoming relatively easy and efficient. All the methods explained above are the classical methods to detect communities. There are many other techniques to detect the community structure e.g. Machine Learning algorithms.

Finally, the community detection is only one aspect to analyze social trends, relationships, and behavior. There are many other means using which we can store, retrieve and analyze this large amount of information.

XI. References

[1] http://nexus.igraph.org/api/dataset_info?id=1&format=html
[2] http://igraph.org/python/doc/igraph.GraphBase-class.html#betweenness
[3] http://igraph.org/python/doc/igraph.clustering.VertexClustering-class.html
[4] http://igraph.org/python/doc/igraph.Graph-class.html
[5] http://snap.stanford.edu/data/egonets-Facebook.html
[6] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821-7826, 2002. [Online]. Available: http://www.pnas.org/content/99/12/7821.full
[7] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, pp. 10008, 2008. [Online]. Available: http://iopscience.iop.org/article/10.1088/1742-5468/2008/10/P10008/meta
[8] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical review E*, vol. 70, no. 6, pp. 0066111, 2004. [Online]. Available: http://journals.aps.org/pre/abstract/10.1103/PhysRevE.70.066111