Web Science cs532-s16

Assignment 5 Report

Dr. Michael L. Nelson

By: Huan Huang
03/03/2016

Problem 1

We know the result of the Karate Club (Zachary, 1977) split. Prove or disprove that the result of split could have been predicted by the weighted graph of social interactions. How well does the mathematical model represent reality?

Generously document your answer with all supporting equations, code, graphs, arguments, etc.

Useful sources include:

* Original paper

http://aris.ss.uci.edu/ lin/76.pdf

* Slides

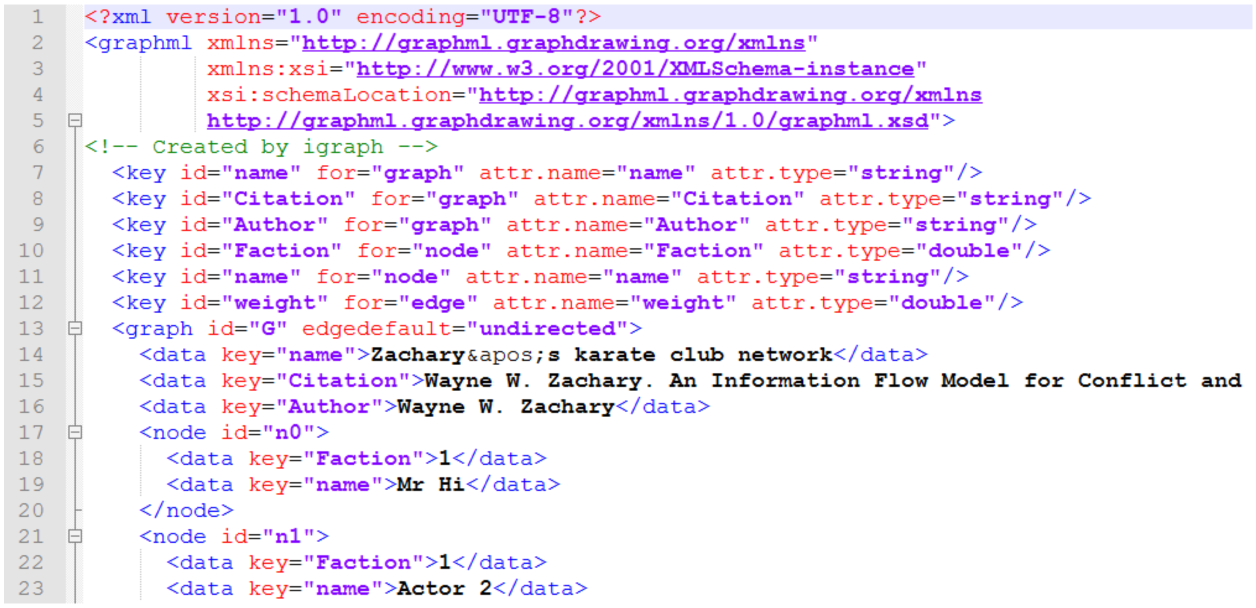
```
http://www-personal.umich.edu/~ladamic/courses/networks/si614w06/ppt/lecture18.ppt

\noindent
http://clair.si.umich.edu/si767/papers/Week03/Community/CommunityDetection.pptx
```

* Code and data

```
http://networkx.github.io/documentation/latest/examples/graph/karate_club.html

http://nbviewer.ipython.org/url/courses.cit.cornell.edu/info6010/resources/11notes.ipynb

http://stackoverflow.com/questions/9471906/what-are-the-differences-between-community-de

http://stackoverflow.com/questions/5822265/are-there-implementations-of-algorithms-for-c

http://konect.uni-koblenz.de/networks/ucidata-zachary

http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm#zachary

https://snap.stanford.edu/snappy/doc/reference/CommunityGirvanNewman.html

http://igraph.org/python/doc/igraph-pysrc.html#Graph.community_edge_betweenness
```

Answer

It is very easy to solve this problem, so long you can find the right data source and tools. I found my data from a Nexus web page `http://nexus.igraph.org/api/dataset_info?id=1&format=html`. This page provides well designed plot-able files which contain the members of the club(name), connections(edges), weight of the edges, groups the members belong to after the split(factions). All the data and relations in these files are based on Zachery's study of the Karate Club.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns
         http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
<!-- Created by igraph -->
  <key id="name" for="graph" attr.name="name" attr.type="string"/>
  <key id="Citation" for="graph" attr.name="Citation" attr.type="string"/>
  <key id="Author" for="graph" attr.name="Author" attr.type="string"/>
  <key id="Faction" for="node" attr.name="Faction" attr.type="double"/>
  <key id="name" for="node" attr.name="name" attr.type="string"/>
  <key id="weight" for="edge" attr.name="weight" attr.type="double"/>
  <graph id="G" edgedefault="undirected">
    <data key="name">Zachary&apos;s karate club network</data>
    <data key="Citation">Wayne W. Zachary. An Information Flow Model for Conflict and
    <data key="Author">Wayne W. Zachary</data>
    <node id="n0">
      <data key="Faction">1</data>
      <data key="name">Mr Hi</data>
    </node>
    <node id="n1">
      <data key="Faction">1</data>
      <data key="name">Actor 2</data>
```

Figure 1: Sample of the data file karate.GraphMl

Now, I have the right data source, I just have to use the right tool to plot it. The plotting module I decided to use is python-igraph. It is designed to handle jobs like this problem. To plot the graph, load the data, decide the layout, and set the options in the actual plotting command. For all of the graphs in problem 1, I used circular layout, I found it easier to do comparison with circular layout. Since we are looking for the relationships of the club members, the "name" is use to set as vertexes, and each vertex is label with the attribute of corresponding "name" key. Here are the sources where I learned how to use python-igraph: `http://igraph.org/python/doc/tutorial/tutorial.html`, A Stackoverflow page, `http://igraph.org/python/doc/igraph.Graph-class.html`.

Next, I have to identify the groups after the split. Since the data file already given each individual the faction they split into, therefore, I can identify the two groups by giving the members different colors based on their faction number. Faction 1 is red, faction 2 is blue.

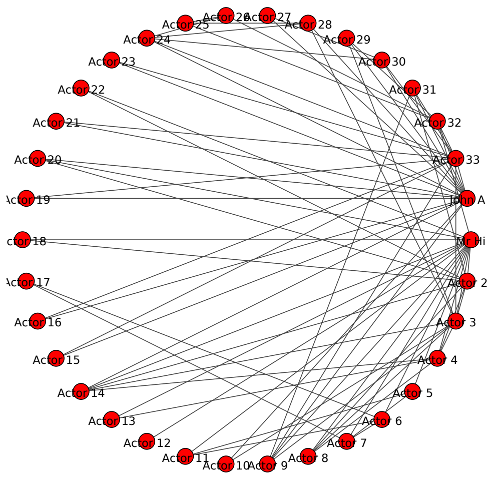


Figure 2: Karate Club before the split

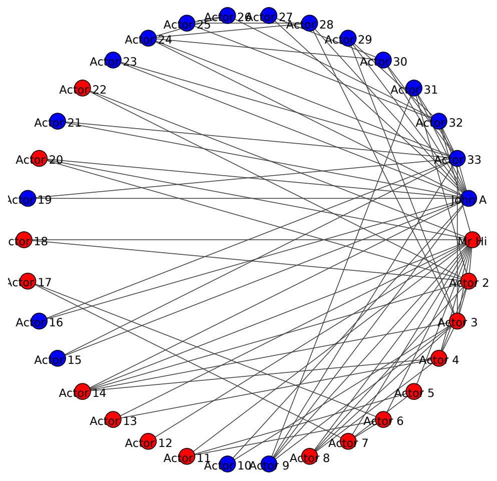


Figure 3: Karate club after the split

The graphs above shows the actual separation of the karate club which is according to Zachery's study. For the next part of this assignment, I will use community detection algorithms to predict the outcome of the split based on the edges and weight of the edges.

Of all the community detection algorithms in python-igraph, Girvan Newman's Edgebetweenness and Leading Eigenvectors are the only 2 algorithms that let me decide how many groups the members split into by setting the cluster number. There, I decide to try both of them.

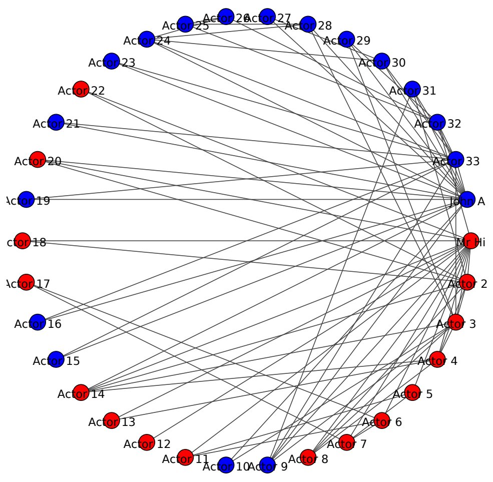


Figure 4: The actual split

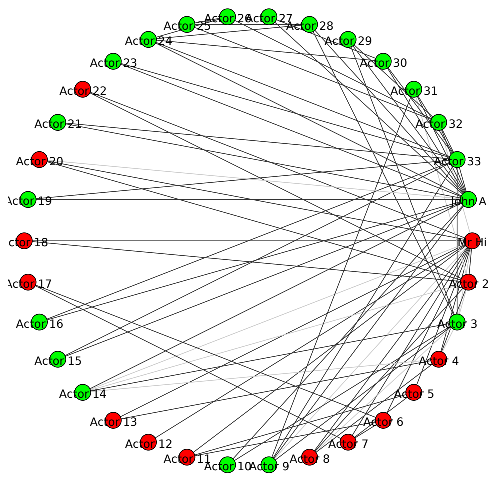


Figure 5: Prediction of Edgebetweenness

Newman's Edgebetweenness did a pretty good job of predicting the split. It got almost every thing right except actor 3 and 14.

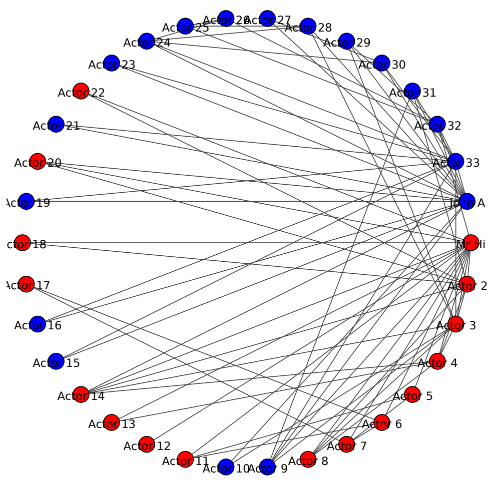


Figure 6: The actual split

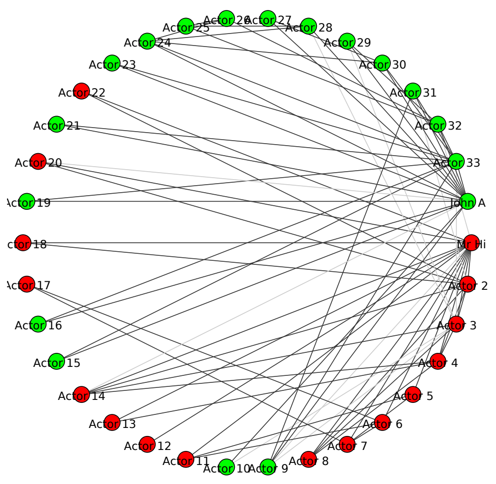


Figure 7: Prediction of Leading Eigenvectors

As you can see, Newman's Leading Eigenvectors was able to correctly predict the split 100%.

Base on the results of the two algorithms, it is safe to say that the result of the karate club split could have been predicted by the weighted graph of social interactions.

```python
#!/usr/bin/env python

from igraph import *

data = load('karate.GraphML')

layout1 = data.layout('circle')
layout2 = data.layout('kk')
plot(data, "beforesplit.pdf", layout=layout1, vertex_label=data.vs['name'])

color = {1: 'red', 2: 'blue'}
plot(data, "aftersplit.pdf", vertex_color = [color[Faction] for Faction in
    data.vs["Faction"]], layout = layout1, vertex_label=data.vs['name'])

graph3 = data.community_edge_betweenness(clusters=2, weights=data.es['weight'
    ])
cluster2 = graph3.as_clustering()
plot(cluster2, "betweenness2.pdf", vertex_label=data.vs['name'], layout=
    layout1)

graph4 = data.community_edge_betweenness(clusters=3, weights=data.es['weight'
    ])
cluster3 = graph4.as_clustering()
plot(cluster3, "betweenness3.pdf", vertex_label=data.vs['name'], layout=
    layout2)

graph5 = data.community_edge_betweenness(clusters=4, weights=data.es['weight'
    ])
cluster4 = graph5.as_clustering()
plot(cluster4, "betweenness4.pdf", vertex_label=data.vs['name'], layout=
    layout2)

graph6 = data.community_edge_betweenness(clusters=5, weights=data.es['weight'
    ])
cluster5 = graph6.as_clustering()
plot(cluster5, "betweenness5.pdf", vertex_label=data.vs['name'], layout=
    layout2)

graph7 = data.community_leading_eigenvector(clusters=2, weights=data.es['
    weight'])
plot(graph7, "eigenvector2.pdf", vertex_label=data.vs['name'], layout=layout1)

plot(data, "aftersplitkk.pdf", vertex_color = [color[Faction] for Faction in
    data.vs["Faction"]], layout = layout2, vertex_label=data.vs['name'])
```

Problem 2

We know the group split in two different groups. Suppose the disagreements in the group were more nuanced – what would the clubs look like if they split into groups of 3, 4, and 5?

Answer

To solve this problem, I just had to change the clusters number to get splits of 3 groups, 4 groups, and 5 groups. I did attempt to use Leading Eigenvectors algorithm at first, since it is more accurate. However, for this particular set of vertexes and relations, Leading Eigenvectors algorithms was not able to give sensible predictions after 3 splits. Therefore, I proceeded with Edgebetweenness algorithm. Also, to show the splits more clearly, I applied the Kamada-Kawai force-directed algorithm(kk) layout.

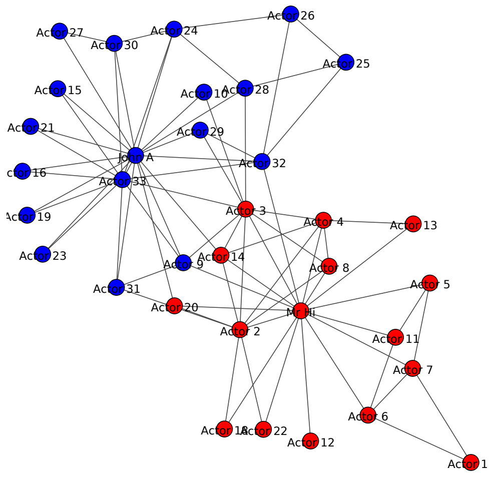


Figure 8: The actual split

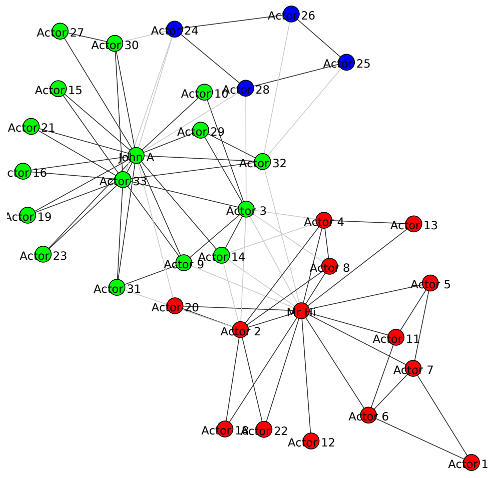


Figure 9: Club splits into 3 groups

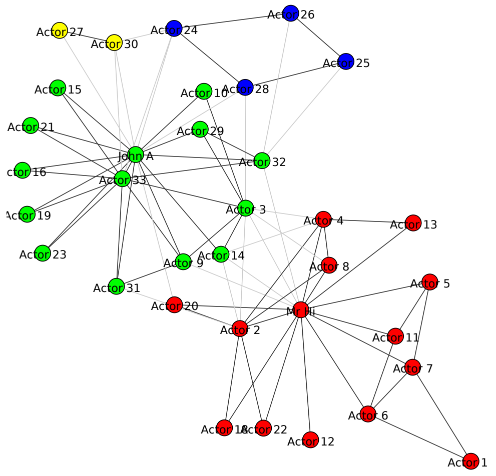


Figure 10: Club splits into 4 groups

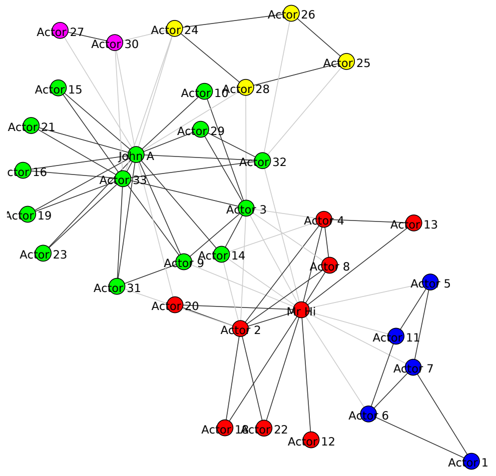


Figure 11: Club splits into 5 groups