Assignment 5

CS532-s16: Web Sciences

Spring 2016

John Berlin

Generated on March 3, 2016

1

Question

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

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-detect

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

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

Before proving whether or not the result of split could have been predicted, let us first understand the problem. Zachary's Karate Club [4] is a very well known graph problem, where the social interactions between two groups "*Mr. Hi* and *John A*", caused a split in that social community. The club as seen in figure 1 is the original club with the results of the split shaded according to which faction each member went too. As seen in figure 1, the graph(club) is highly connected, which leaves us to wonder how would we figure out how to predict the division solely using the graph itself.

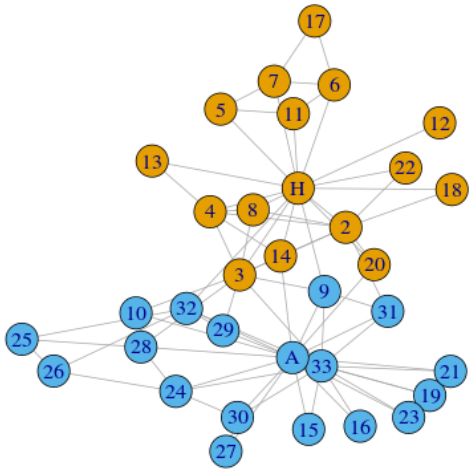

Figure 1: Zachary's Karate Club

To determine if the graph structure has enough information to predict the split, I used the edge betweenness algorithm as described by Girvan, Newman in their paper Community structure in social and biological networks [1]. In that paper they stated, that the "betweenness of an edge as the number of shortest paths between pairs of vertices that run along it". Furthermore they state that "edges connecting communities will have high edge betweenness" and "by removing those edges" the communities are exposed. Now to take from Dr. Nelson's lecture #6 "Edge betweenness: Total amount of flow an edge carries

between all pairs of nodes where a single unit of flow between two nodes divides itself evenly among all shortest paths between nodes". I believe that is straight from the Girvan Newman paper when they generalize Freeman's betweenness centrality[1].

This algorithm as stated in the paper by Girvan, Newman is

1. Calculate the betweenness for all edges in the network
2. Remove the edges with the highest betweenness
3. Recalculate betweenness for all edges affected by the removal
4. Repeat from step 2 until no edges remain

In using this algorithm to find the split, I must add one caveat to step number 4, which is to stop when the number of communities desired has been reached. To know when that stopping point has been reached is easy, simply count the number of connected components in the graph. The connected components at the start of the algorithm will be one, as the original graph is one piece. When the algorithm has removed enough edges the number connected components will be increased; in short I am counting the number of connected subgraphs [3].
I implemented this algorithm in Python using the Networkx library and can be seen in listing 1 produced the split seen in figure 3. Please see the comments for further details. But stated simply the program runs as such

1. Calculate the betweenness for all edges in the network
2. Get the maximum value from the dictionary gotten from edge_betweenness_centrality
3. Iterate through the returned dictionary and remove the edges whoes score is equal to max. This is to account for ties if any
4. Repeat until we reach the desired number of communities denoted splitTo by checking the number of connected components in the graph

The results of my implementation can be seen in table 1. The actor column represents the member number from the club with actor 1 being Mr. Hi and actor 34 being John A. The following two columns show the faction and result of the split from the Zachary paper [4]. The final two columns show the faction membership as perdicted my Girvan-Newman implementation and if it matched membership correctly, hit for yes and miss for no.

From the Zachary paper his model had a 97% percent accuracy rating missing only 1 membership out of the 33. Whereas my model missed 4 out of the 34 resulting in a 88% accuracy score. In comparison to the work done by Zachary I as well missed the faction that actor number 9 would go too. But it also must be noted that Girvan-Newman used the unweighted version of the Zachary graph.

This model as produced by the python implementation seemed to be off, so

I decided to use R to see if I did anything wrong. Why because as I went looking into graph libraries the igraph R package kept coming up so I decided to use it.

The code used in my R implementation can be found in listing 3 and the resulting graph split in figure 2. Now this implementation is the most interesting. I used the igraph library and consulted the pdf reference that accompanied it on cran[2]. In the library is this method called cluster_edge_betweenness which returns to us a communities object. This communities class performs the Girvan Newman algorithm for us and returns to us the edges removed, their betweenness score and much more. The comments in the code contain details on what is happening.

Since I already did the bigger analysis for my python implementation I will briefly break down the improvements. The r version correctly said the actor 9 would go to the Officers faction, actor number 14 was incorrectly placed in the Officers faction but actor 22 was correctly placed in Mr. Hi's faction. With these improvements I now miss only 2 out of the 34 resulting in a 94% accuracy score.
By this I can safely say that the split can be predicted with 90% accuracy of the actual split.

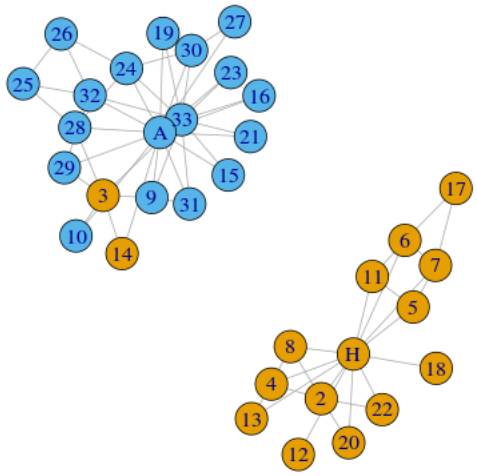

Figure 2: Club split by R implementation of Girvan Newman

```
import networkx as nx

def gnewman(club,splitTo = 2):
    itteration = 0
    # ok so why do I check the number of connected components
    # for an undirected graph it is know that a connected component of an
    # an undirected graph is a subgraph in which any two vertices are connected to each other by paths
    # this is useful for this application since we are splitting a graph into two subgraphs
    # ie to mathematically represent the splitting of the club
    while nx.number_connected_components(club) < splitTo:
        # returns to us edges with the weights
        between = nx.edge_betweenness_centrality(club,normalized=False)
        # we want the edges with the highest edge betweenness centrality
        # there might be ties so just get the max betweenness
        m = max(between.values())
        # unpack the tuple returned to us by between.items ((u,v),maxBetweenScore)
        for (hU,hV),val in between.items():
            # check to see if m(max betweenness score) is equal to val
            # removes ties along the way
            if val == m:
                club.remove_edge(hU,hV)
                print("removed edge %s—%s with betweenness score of %f"%(hU,hV,m))
        itteration += 1

        print("————————————————————————")
        # this print out can be uncommented it simply shows the same metric as described two different ways
        # print(nx.number_connected_components(club),len(list(nx.connected_component_subgraphs(club))))
    print("total iterations %d for splitting into %d"%(itteration,splitTo))

if __name__ == "__main__":

    for splitTo in range(2,6):
        #this is a weighted version of the graph from http://www.nexus.igraph.org/api/dataset_info?id=1&format=html
        karateClub = nx.read_graphml("karate.GraphML")
        # karateClub = nx.read_gml("nxKclub2.gml")
        # karateClub = nx.karate_club_graph()
        gnewman(karateClub,splitTo)
        print("————————————————————————————\n")
        nx.write_dot(karateClub,"kc%d.dot"%splitTo)
```

Listing 1: Girvan Newman edge betweenness centrality

Actor	Faction Membership From Data	Zachary Membership From Split	Girvan-Newman Faction Membership From Split	Hit Miss for my Girvan-Newman
1	Mr. Hi	Mr. Hi	Mr. Hi	Hit
2	Mr. Hi	Mr. Hi	Mr. Hi	Hit
3	Mr. Hi	Mr. Hi	Officer	Miss
4	Mr. Hi	Mr. Hi	Mr. Hi	Hit
5	Mr. Hi	Mr. Hi	Mr. Hi	Hit
6	Mr. Hi	Mr. Hi	Mr. Hi	Hit
7	Mr. Hi	Mr. Hi	Mr. Hi	Hit
8	Mr. Hi	Mr. Hi	Mr. Hi	Hit
9	Mr. Hi	Officers	Officers	Miss
10	Officers	Officers	Mr. Hi	Miss
11	Mr. Hi	Mr. Hi	Mr. Hi	Hit
12	Mr. Hi	Mr. Hi	Mr. Hi	Hit
13	Mr. Hi	Mr. Hi	Mr. Hi	Hit
14	Mr. Hi	Mr. Hi	Mr. Hi	Hit
15	Officers	Officers	Officers	Hit
16	Officers	Officers	Officers	Hit
17	Mr. Hi	Mr. Hi	Mr. Hi	Hit
18	Mr. Hi	Mr. Hi	Mr. Hi	Hit
19	Officers	Officers	Officers	Hit
20	Mr. Hi	Mr. Hi	Mr. Hi	Hit
21	Officers	Officers	Officers	Hit
22	Mr. Hi	Mr. Hi	Officers	Miss
23	Officers	Officers	Officers	Hit
24	Officers	Officers	Officers	Hit
25	Officers	Officers	Officers	Hit
26	Officers	Officers	Officers	Hit
27	Officers	Officers	Officers	Hit
28	Officers	Officers	Officers	Hit
29	Officers	Officers	Officers	Hit
30	Officers	Officers	Officers	Hit
31	Officers	Officers	Officers	Hit
32	Officers	Officers	Officers	Hit
33	Officers	Officers	Officers	Hit
34	Officers	Officers	Officers	Hit

Table 1: Comparisons of the results of split as generated by my Girvan Newman implementation to Zachary's predictions and the actual data as was done in the original paper

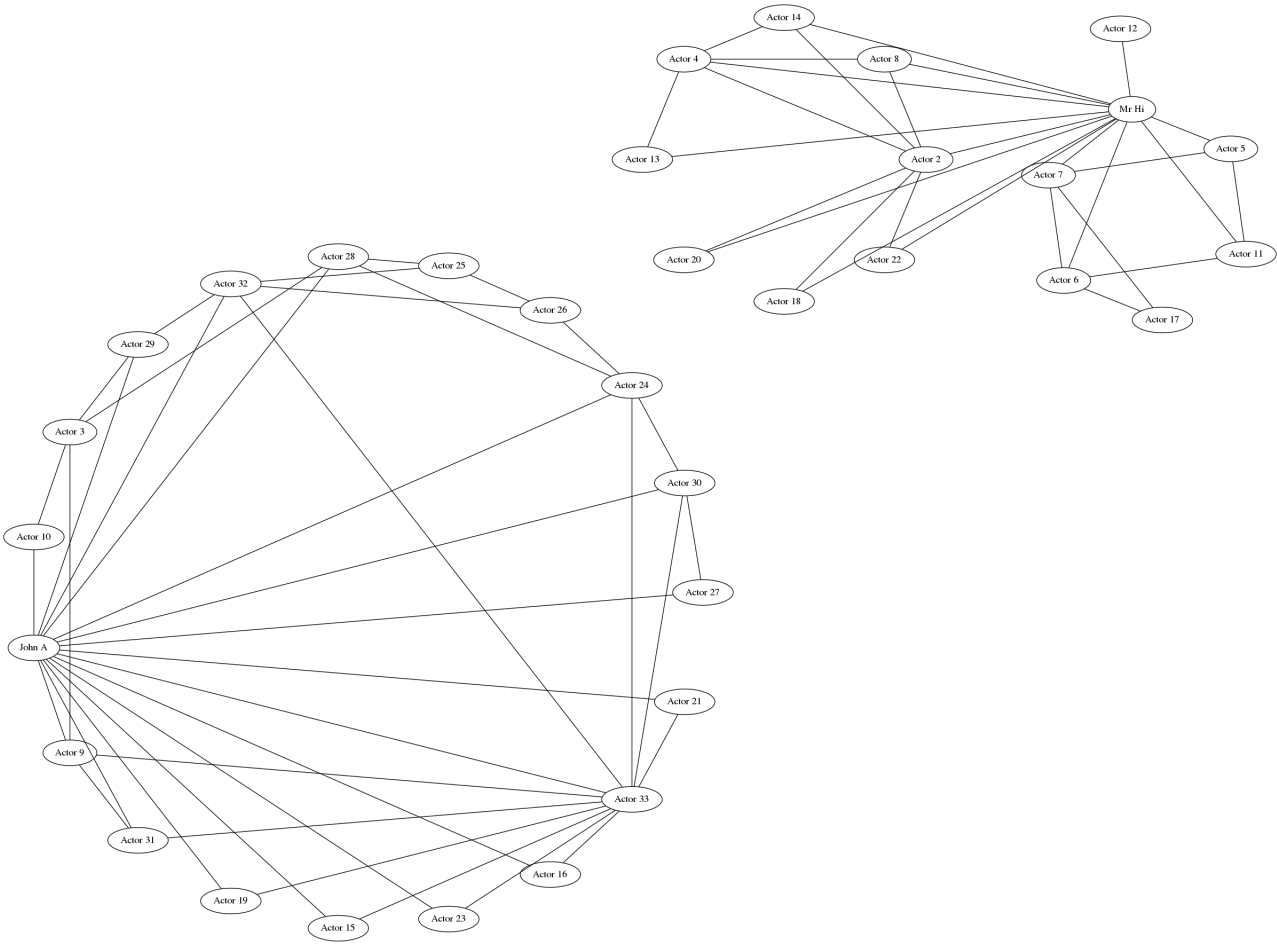

Figure 3: Club split by my implementation of Girvan Newman

```
$ circo -Tpng -o kc.png kc2.dot
$ circo -Tpng -o kc3.png  kc3.dot
$ circo -Tpng -o kc4.png  kc4.dot
$ circo -Tpng -o kc5.png  kc5.dot
```

Listing 2: Convert dot files to png

```
library(igraph)
library(igraphdata)

#ok so lets do this in R
data(karate)
karateClub <- karate
#hmmm this looks interesting
#edge.betweenness.community(cluster_edge_betweenness),65 from the pdf on cran
ceb <- cluster_edge_betweenness(karateClub)
#what does this give us a communities class
print(class(ceb))
#look what we have here! it removed edges for us well sorta
print(ceb$removed.edges)
#this will print edge betweenness ;)
print(ceb$algorithm)

#set to true to see debug output
debug <- TRUE

#how many communities do we wish to split to
splitTo <- 2

#ok some set up
#want to preserve the final graph that we create
# I will not be touching the original karate club
finalPlot <- karateClub
plot.igraph(finalPlot)

#ok so I want have a counter to keep track
#of where I am in the algorithm
removeIndex <- 1
#use the same scheme as python I want to make sure
#the number of connected componets are less than 2
#for the Zachary Analysis
numConnected <- count_components(karateClub)

#loop
while(numConnected < splitTo){
  #ok so this feels like cheating here
  #the scheme is to progressively remove edges the ebc communities class calculated for us
  #if you start at index 0 you get nothing so I start at 1 to removeIndex -1
  #base removeIndex=1 seq(1,removeIndex-1) yeild 1 0
  #so the ebc$removed.edges[seq(1,removeIndex-1) ] yeild edge at pos 1 :)
   deletedG <- delete.edges(karateClub, ceb$removed.edges[seq(1, removeIndex-1)])
  #deleteG is now the result of iteration N of Girvan Newman

  #increment our iteration of Girvan Newman up by one
   removeIndex <- removeIndex + 1
  # plot.igraph(deletedG)
   if(debug){
     print(seq(1,removeIndex-1))
     print(ceb$edge.betweenness[seq(1,removeIndex-1)])
```

```
    print(ceb$removed.edges[seq(1,removeIndex-1)])
    print("++++++++++++++++++++++++++++++")
  }
  #get the number of connected componets ie
  #have we split the graph into two subgraphs
  numConnected <- count_components(deletedG)
  finalPlot <- deletedG
}

plot.igraph(finalPlot)
```

Listing 3: Girvan Newman edge betweenness centrality in R

2

Question

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

Using the same R code as in the second implementation I changed the splitTo value to 3 communities figure 4,4 communities figure5, and 5 communities figure6. Those figures are seen below.

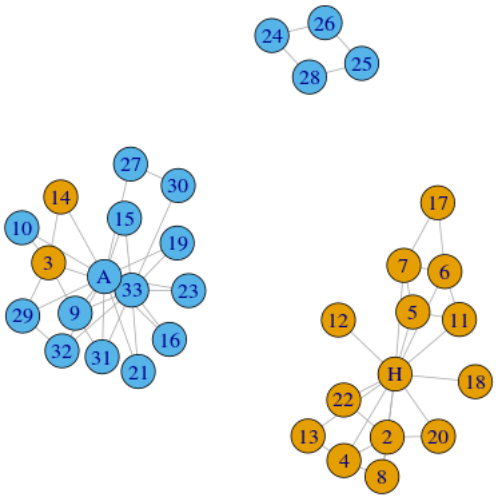

Figure 4: Karate Club Graph Split Into 3 Communities

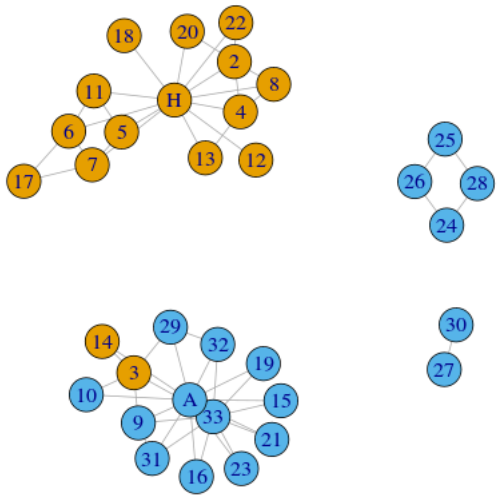

Figure 5: Karate Club Graph Split Into 4 Communities

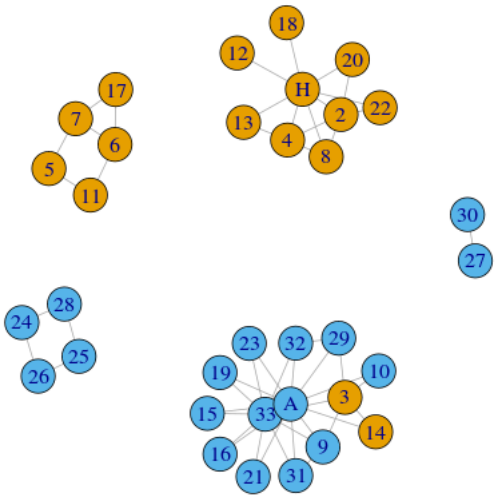

Figure 6: Karate Club Graph Split Into 5 Communities

References

[1] Girvan, M., and Newman, M. E. J. Community structure in social and biological networks. *arXiv:cond-mat/0112110* (2001).

[2] many see authors file. igraph: Network analysis and visualization, 2015.

[3] Wikipedia. Connected component (graph theory) — wikipedia, the free encyclopedia, 2015. [Online; accessed 3-March-2016].

[4] Zachary, W. W. An information flow model for conflict and fission in small groups. *Journal of anthropological research* (1977).