Introduction to Web Science: Assignment #6

Dr. Nelson

Alexander Nwala

Thursday, October 30, 2014

Contents

Problem 1 **3**

Problem 2 **5**

Problem 1

We know the result of the Karate Club (Zachary, 1977) split (Figure 1b). 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/le
cture18.ppt

http://clair.si.umich.edu/si767/papers/Week03/Community/CommunityDetect
ion.pptx
```

Code and data

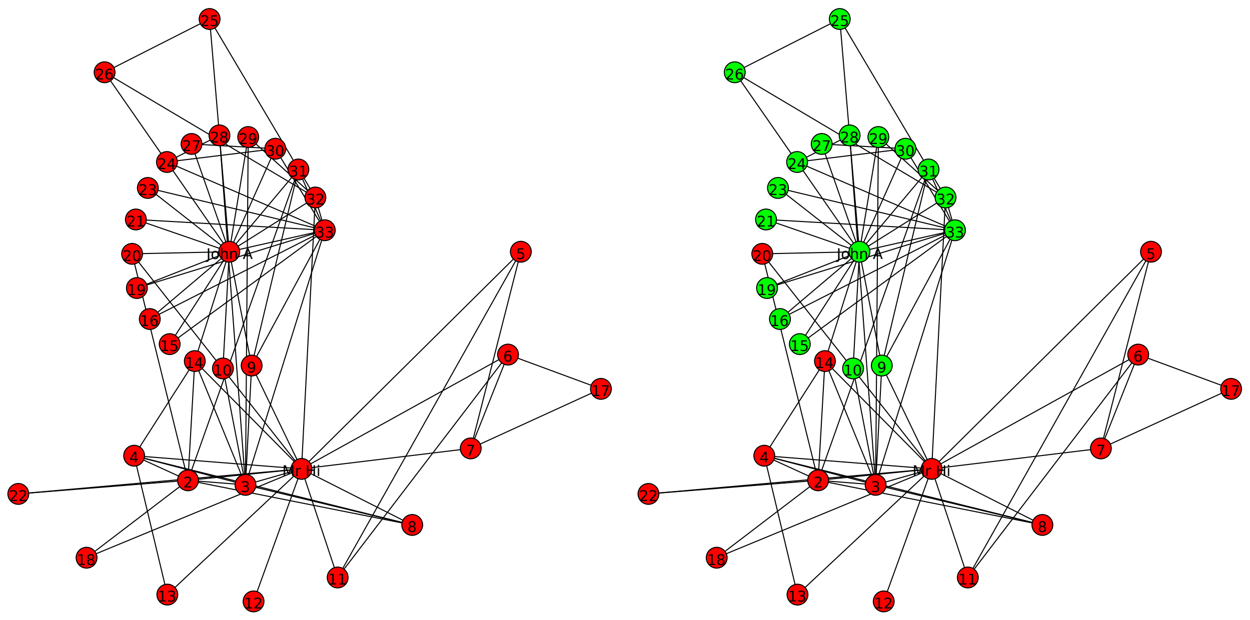
```
http://networkx.github.io/documentation/latest/examples/graph/karate_cl
ub.html

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

http://stackoverflow.com/questions/9471906/what-are-the-differences-bet
ween-community-detection-algorithms-in-igraph/9478989#9478989

http://stackoverflow.com/questions/5822265/are-there-implementations-of
-algorithms-for-community-detection-in-graphs

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

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

SOLUTION 1

The strategy I employed in order to prove the result of the split (**Figure 1b**) is based upon the following rationale: Given the Karate Club graph, unity could be attributed to a single important node. However, if importance is split among more than one node, the consequence of this could be disunity or split. This means we should expect to see multiple clusters as opposed to a single cluster.

Problem 1 continued on next page. . .

Figure 1: Zachary's Karate Club Graphs



(a) Original Graph (b) Result of Split Graph

The mathematical model which mirrors my rationale is the concept of centrality (indicates the most important nodes in a graph). I used the edge betweenness centrality measure in order to measure the centrality. Finally in order to prove the result of the split, I applied the Girvin-Newman algorithm to incrementally see if I could derive the resultant split beginning from the original (non-split) graph (**Figure 1a**). The idea behind using the Girvin-Newman algorithm is, initially, there is unity (1 cluster in the network), then we seek to look for factions or communities in the network by continuously calculating the betweenness and removing the edge with the highest betweenness.

```
    The file karate.GraphML contains the karate club graph
```

The following is a summary of the Girvin-Newman Algorithm I implemented using igraph [1] :

```
Given graph G,
while( edgeCount > 0 and clusterCount < maximumClusterThreshold )
{
    allEdgesBetweennessList = G.getBetweennessValues()
    maximumBetweennessValue = getMaximum(allEdgeBetweennessList)

    edgeWithMaximumBetweennessValue =
    getEgdeWithMaximumBetweennessValue(maximumBetweennessValue)

    G.remove(edgeWithMaximumBetweennessValue)
}
```

Listing 1 is an outline of the implementation

Listing 1: Girvin-Newman Algorithm

```
#Girvin-Newman Algorithm For Community Detection
def girvinNewmanAlgorithm(maximumClusterThreshold):
...
     while karateClubGraph.ecount() > 0 and clusterCount < maximumClusterThreshold...)
         :

          clusters = karateClubGraph.clusters('weak')
          clusterCount = len(clusters)

          visual_style["vertex_color"] = [color_dict_vertices[node['vertex_color']]
              for node in karateClubGraph.vs]

          # calculate the edge betweenesses
          edgeBetweenness = karateClubGraph.edge_betweenness()

          # find the index of the edge with the maximum betweenness:http://lists.
              nongnu.org/archive/html/igraph-help/2008-11/msg00047.html
          indexOfEdgeWithMaximumBetweenness = max(xrange(len(edgeBetweenness)), key =
              edgeBetweenness.__getitem__)

          #modify edge color of edge with maximum betweenness
          karateClubGraph.es[indexOfEdgeWithMaximumBetweenness]['edge_color'] = 'blue'
          karateClubGraph.es[indexOfEdgeWithMaximumBetweenness]['edge_width'] = 5
          drawGraph('social_network' + str(iteration)+ '_' +str(clusterCount))

          #remove edge of maximum betweenness
          karateClubGraph.delete_edges(indexOfEdgeWithMaximumBetweenness)
          iteration = iteration + 1
...
```

CONCLUSION 1

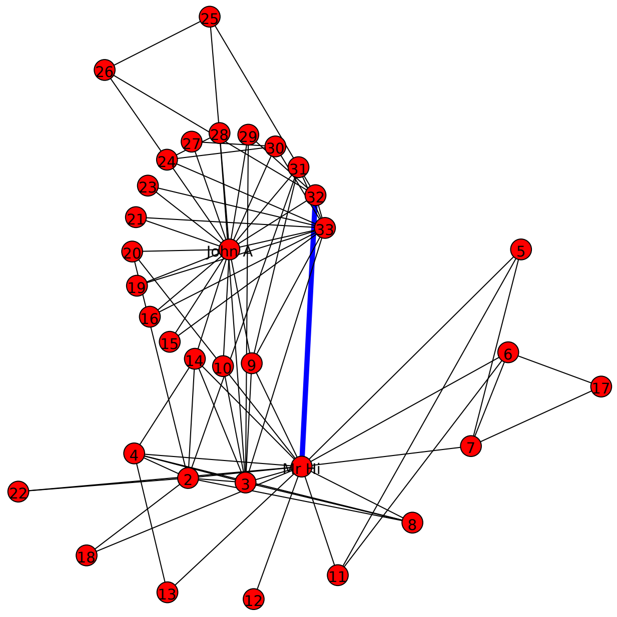
After the 11th iteration (12th run), as seen in **Figure 4d**, the graph contains 2 clusters with John A. as the central figure of the first cluster, and Mr. Hi - the second cluster. Let us compare the factions from **Figure 4d** to the result of the split, **Figure 1b**. Let us call the original result of the split graph RS and the results of running the Girvin-Newman algorithm GV. As seen below, only 1 descrepancey exists - Node 3. So it is fair to say the Girvin-Newman Algorithm fairly represents reality.

```
Faction 1:
  SR:  <9, 10, 15, 16, 19, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33>
  GV: 3<9, 10, 15, 16, 19, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33>
Faction 2:
  SR: <2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 20, 22>
  GV: <2, ., 4, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 20, 22>
```

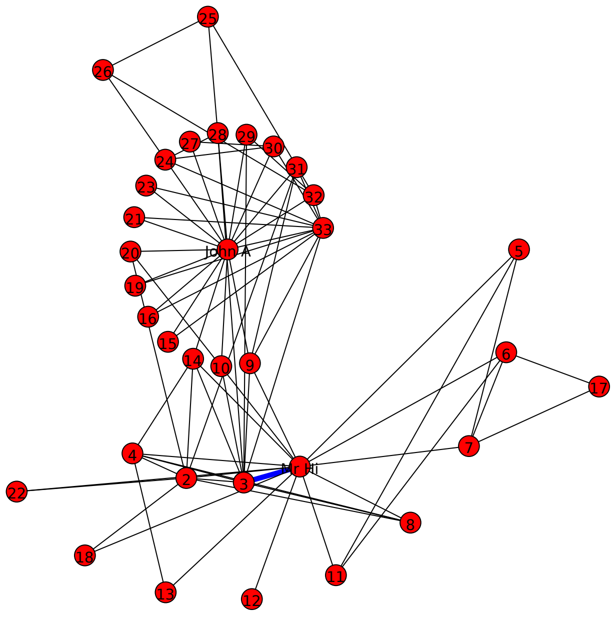
Problem 2

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?

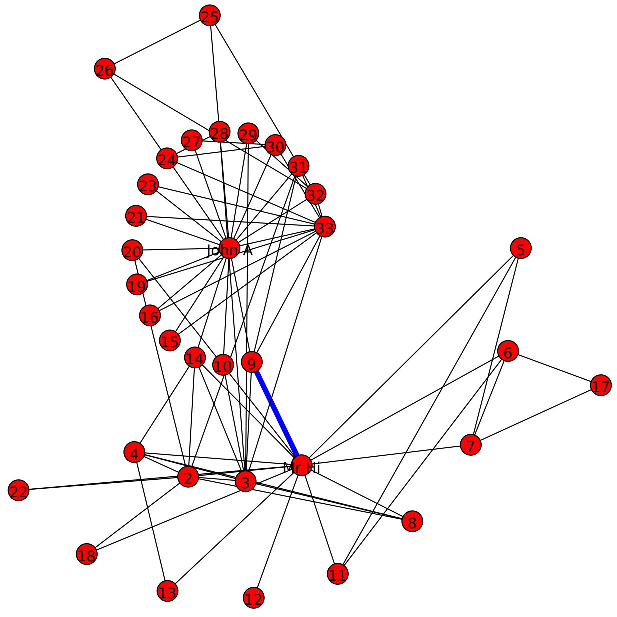
Figure 2: Multiple Iterations Of The Girvin-Newman Algorithm



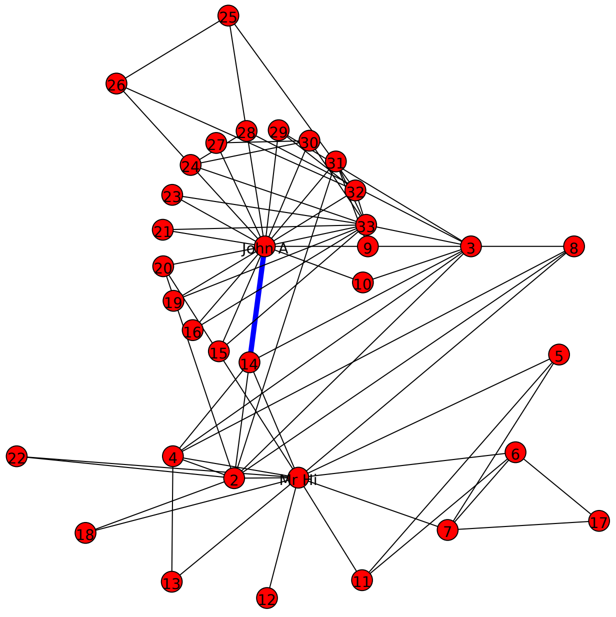
(a) Iteration: 0, Clusters: 1



(b) Iteration: 1, Clusters: 1

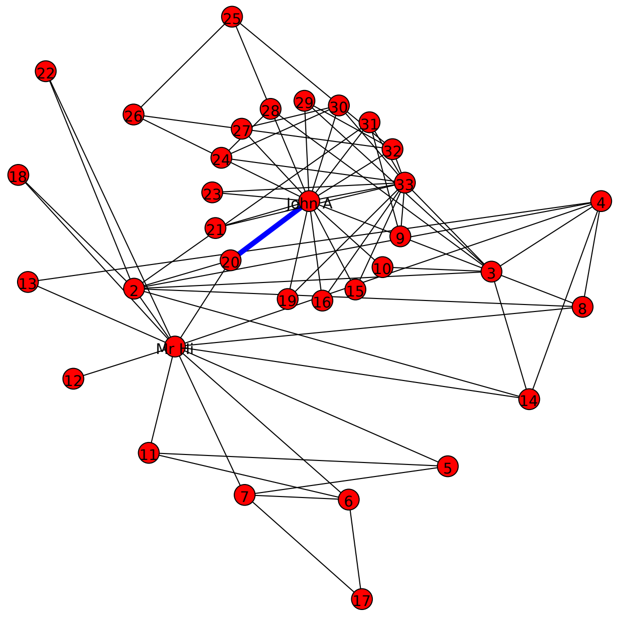


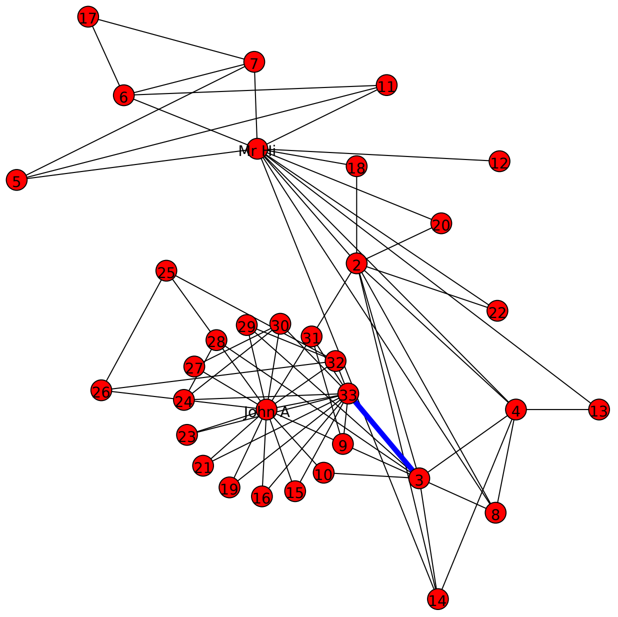
(c) Iteration: 2, Clusters: 1

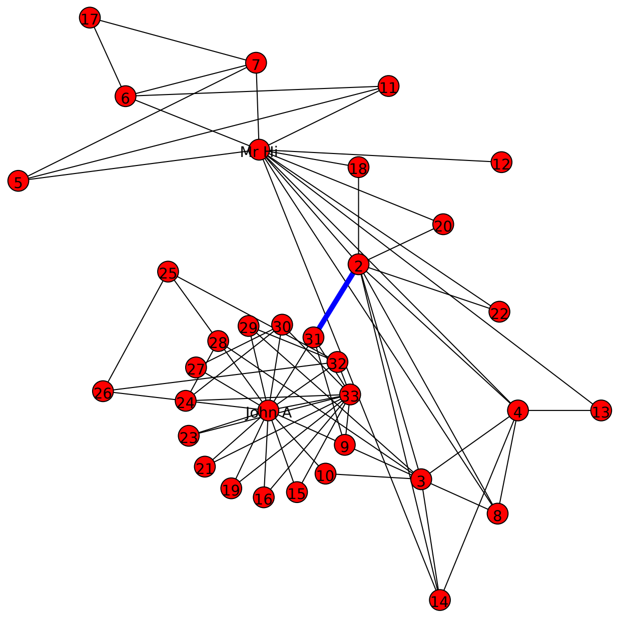


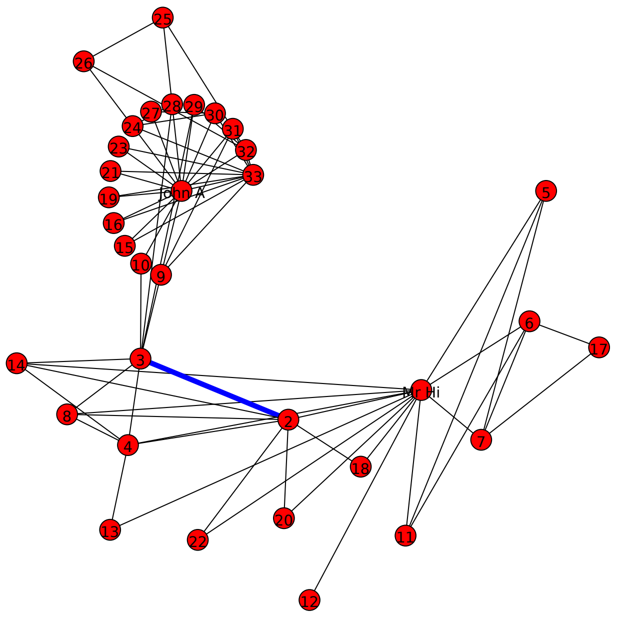
(d) Iteration: 3, Clusters: 1

Figure 3: Multiple Iterations Of The Girvin-Newman Algorithm


(a) Iteration: 4, Clusters: 1

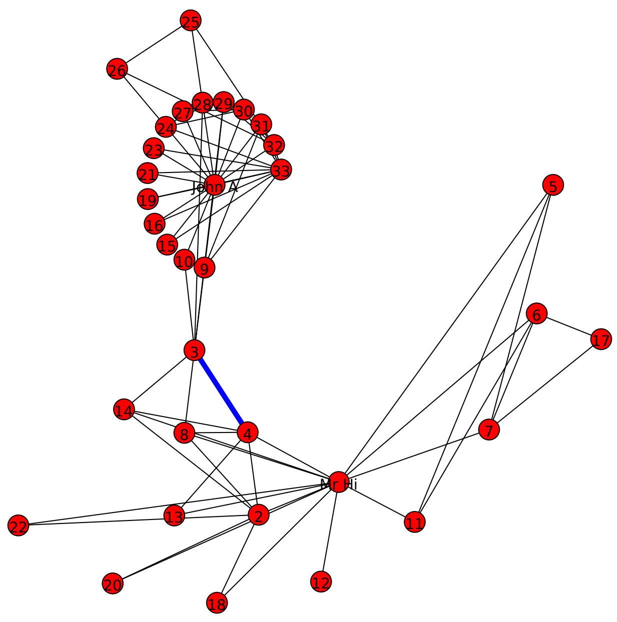

(b) Iteration: 5, Clusters: 1


(c) Iteration: 6, Clusters: 1

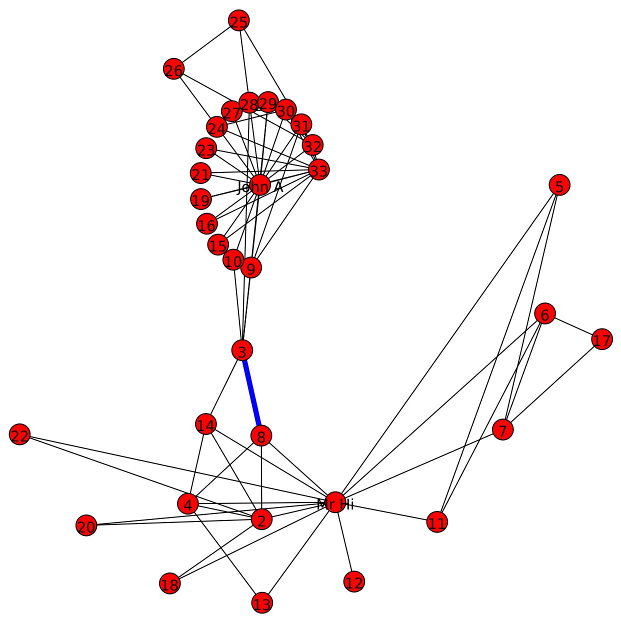

(d) Iteration: 7, Clusters: 1

Problem 2 continued on next page. . .

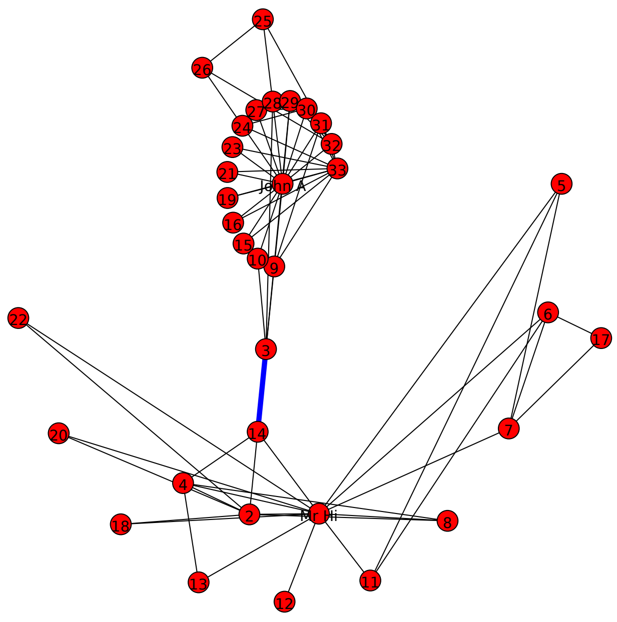
Figure 4: Multiple Iterations Of The Girvin-Newman Algorithm



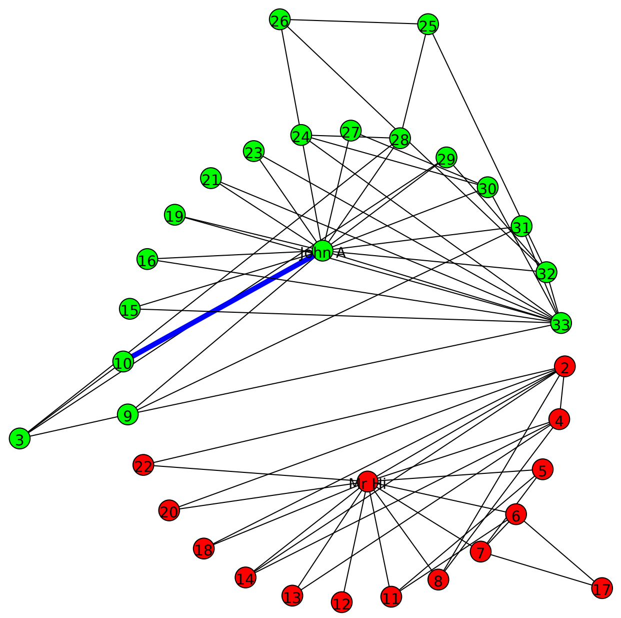
(a) Iteration: 8, Clusters: 1



(b) Iteration: 9, Clusters: 1



(c) Iteration: 10, Clusters: 1



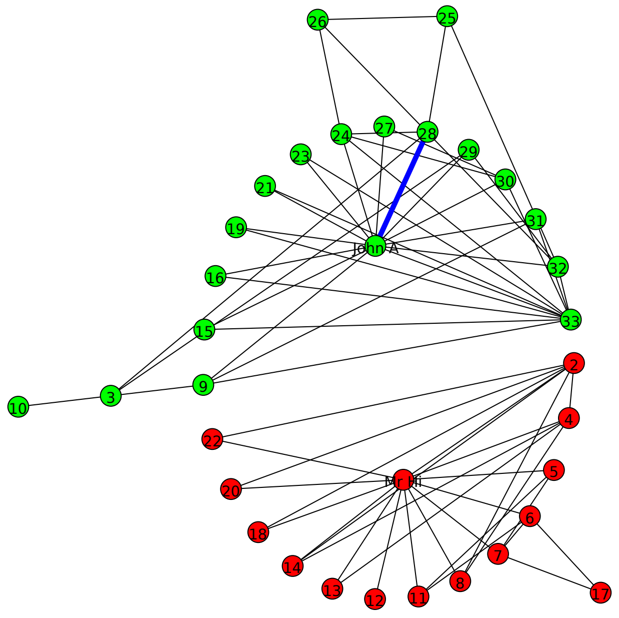
(d) Iteration: 11, Clusters: 2

Problem 2 continued on next page. . .

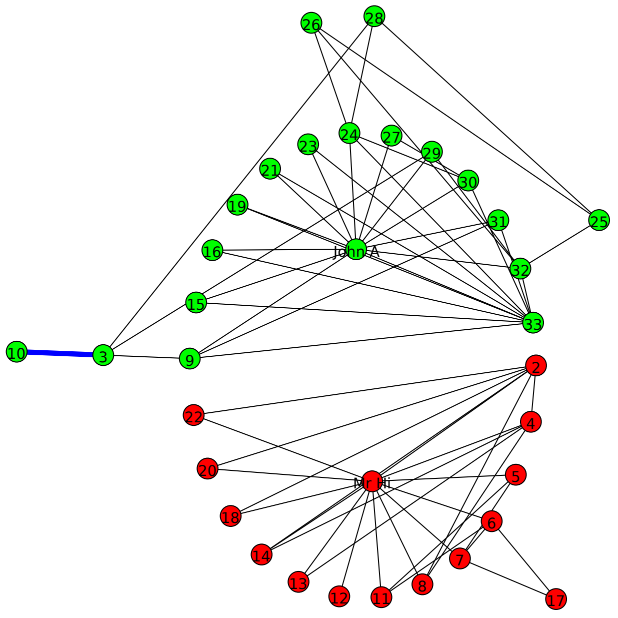
SOLUTION 2

As seen from **Figure 5c**, we have 3 groups after 14 iterations (15 runs)
As seen from **Figure 6c**, we have 4 groups after 18 iterations (19 runs)
As seen from **Figure 8a**, we have 5 groups after 24 iterations (25 runs)

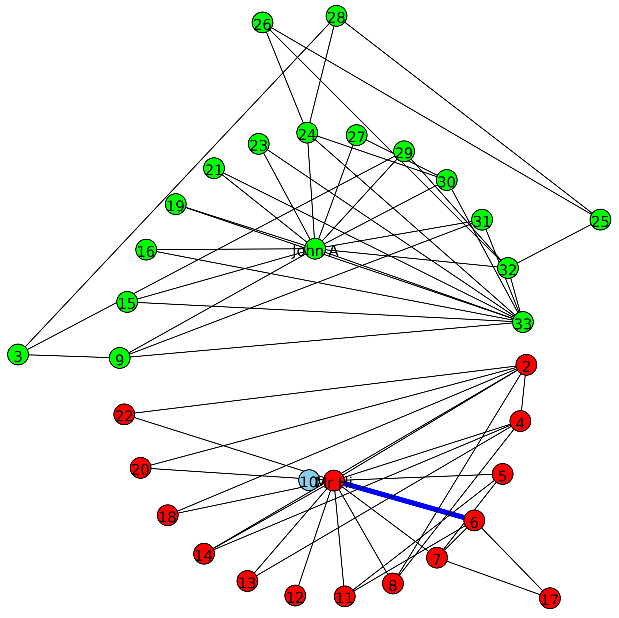
Figure 5: Multiple Iterations Of The Girvin-Newman Algorithm



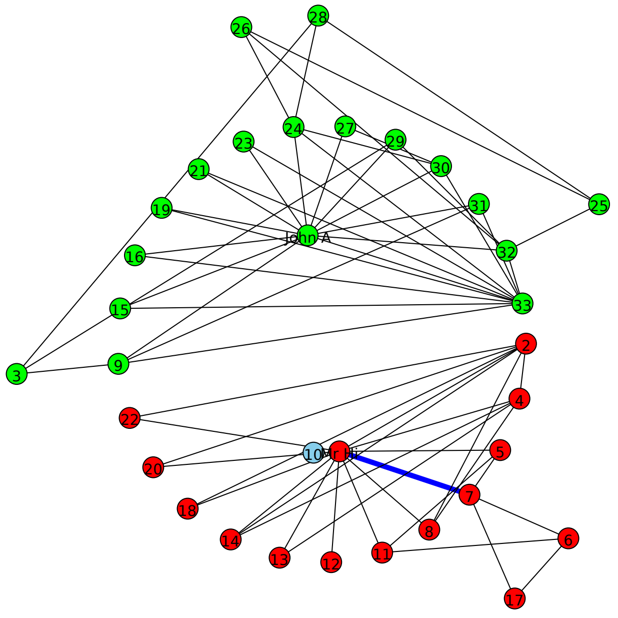
(a) Iteration: 12, Clusters: 2



(b) Iteration: 13, Clusters: 2

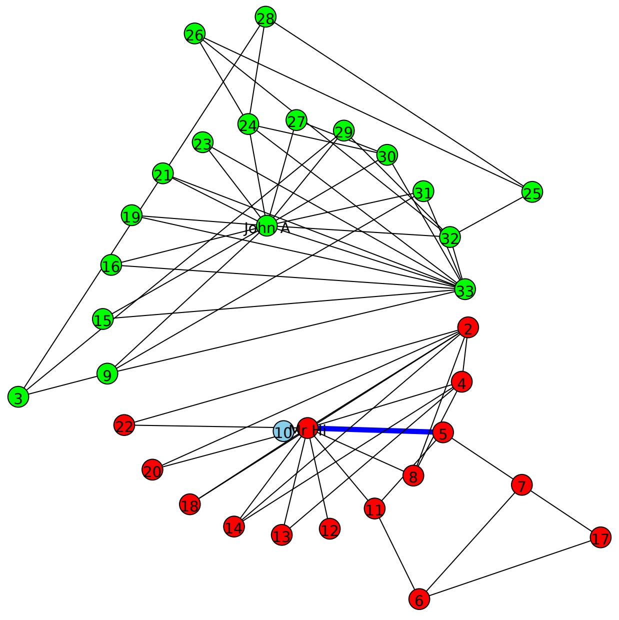


(c) Iteration: 14, Clusters: 3

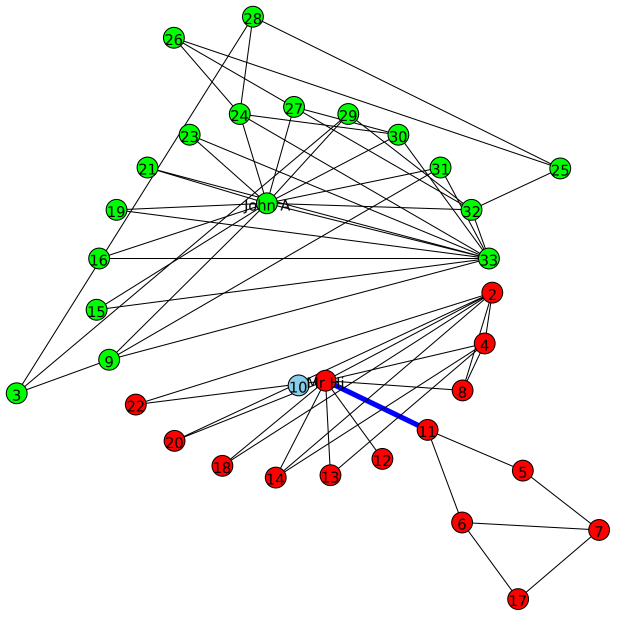


(d) Iteration: 15, Clusters: 3

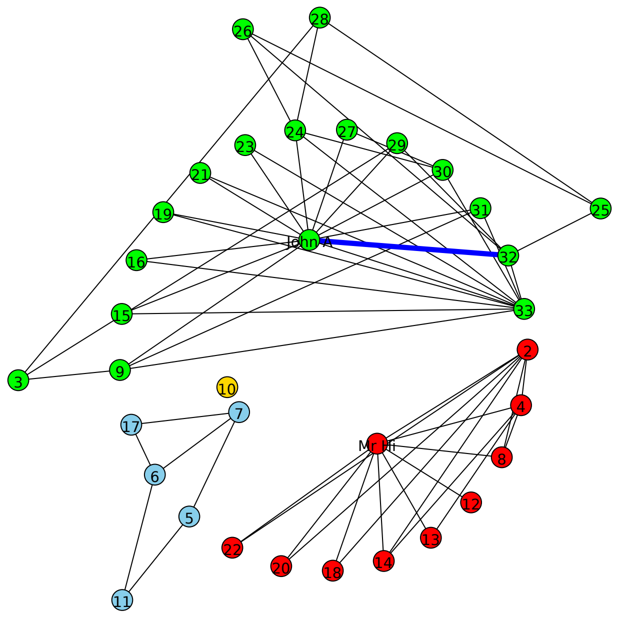
Figure 6: Multiple Iterations Of The Girvin-Newman Algorithm



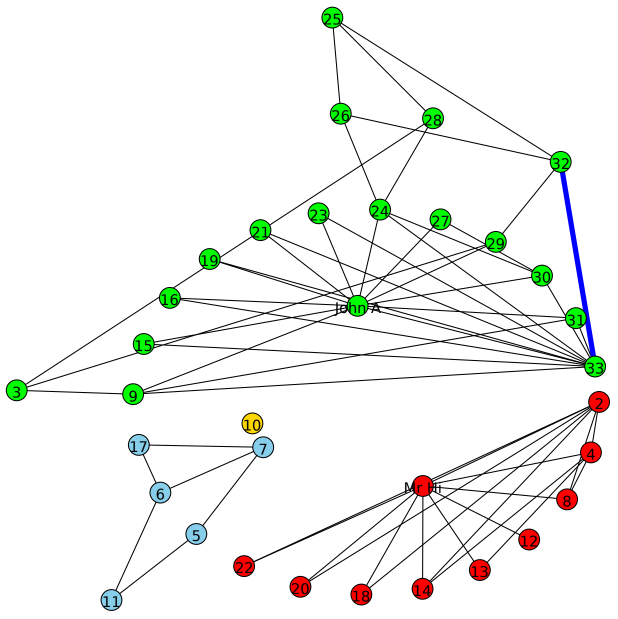
(a) Iteration: 16, Clusters: 3



(b) Iteration: 17, Clusters: 3

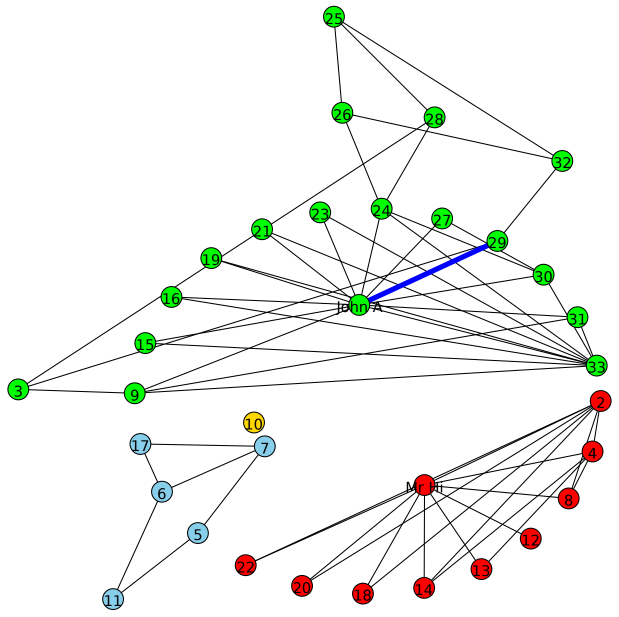


(c) Iteration: 18, Clusters: 4

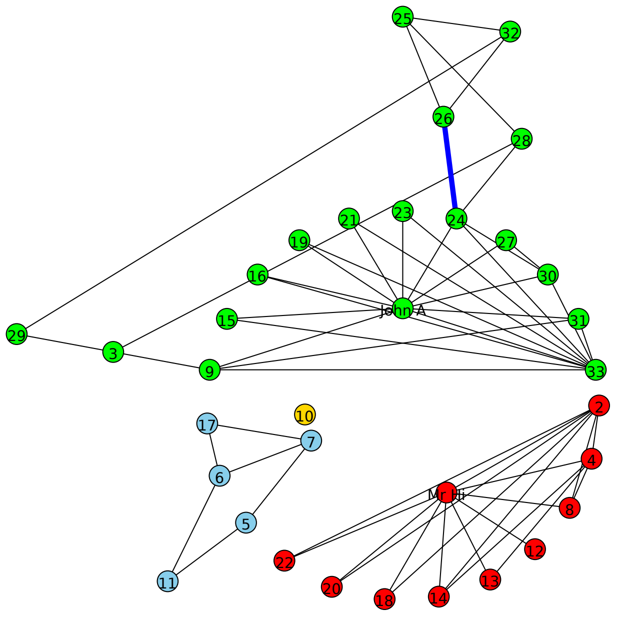


(d) Iteration: 19, Clusters: 4

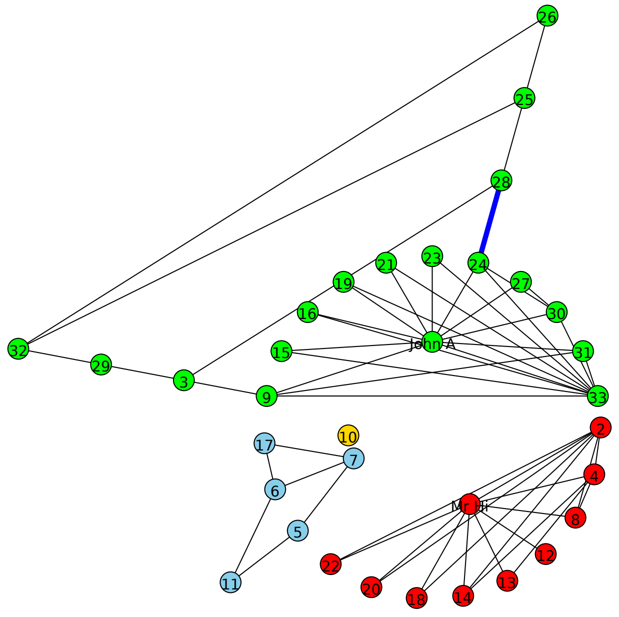
Figure 7: Multiple Iterations Of The Girvin-Newman Algorithm



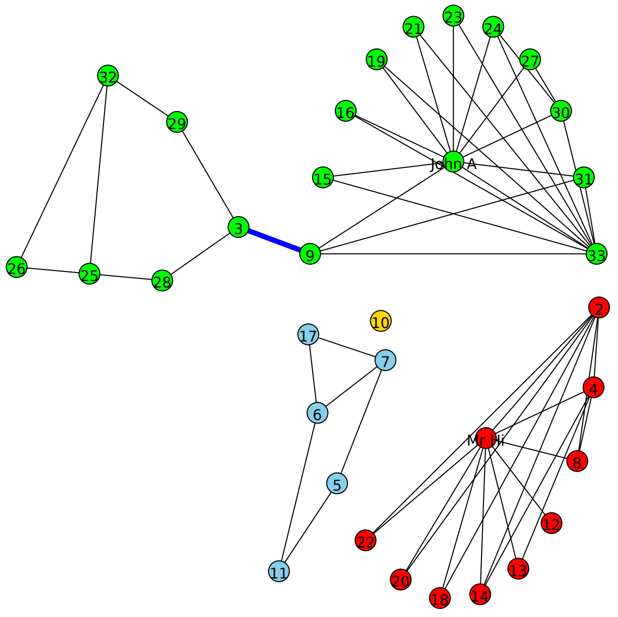
(a) Iteration: 20, Clusters: 4



(b) Iteration: 21, Clusters: 4

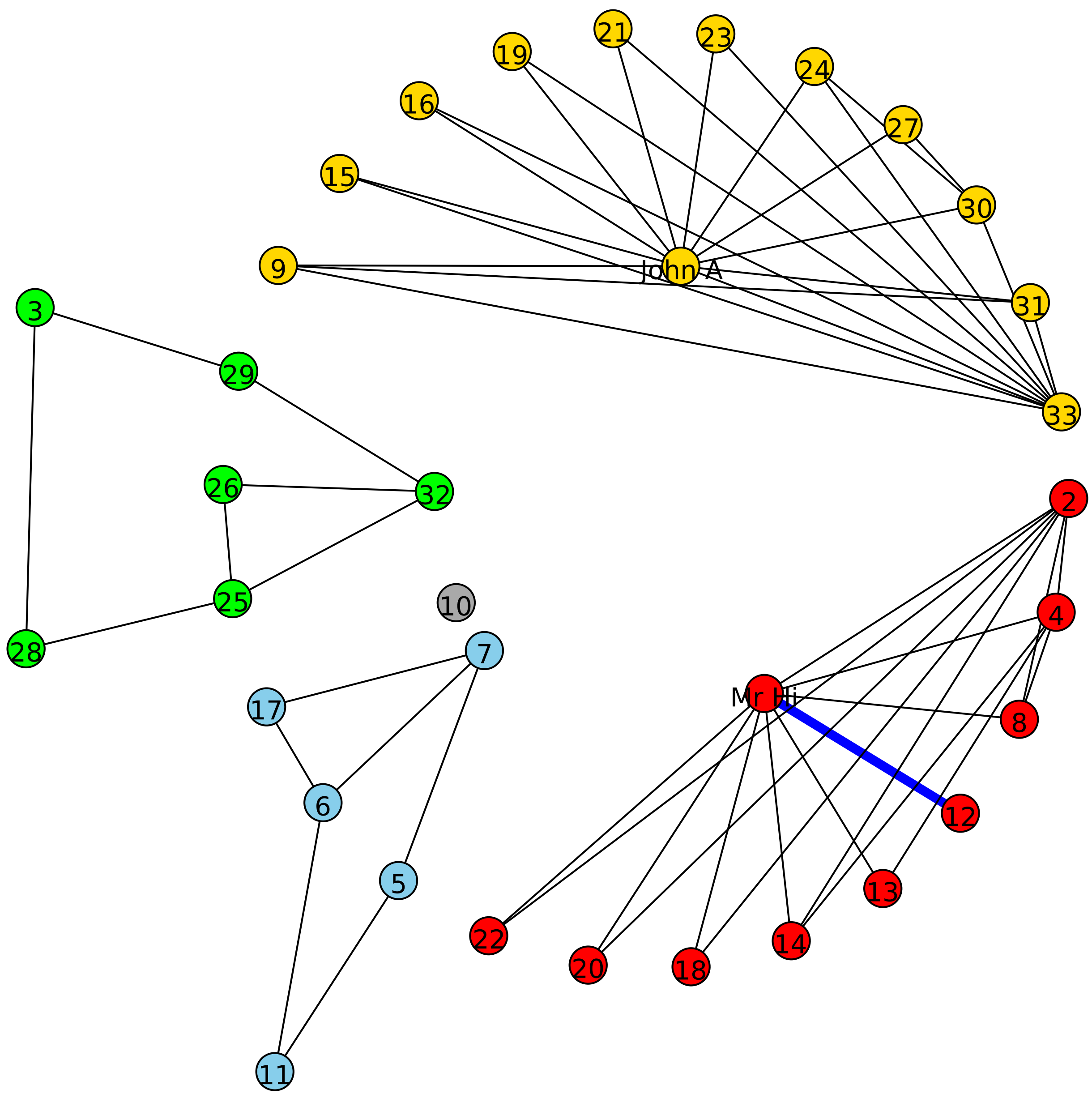


(c) Iteration: 22, Clusters: 4



(d) Iteration: 23, Clusters: 4

Figure 8: Multiple Iterations Of The Girvin-Newman Algorithm



(a) Iteration: 24, Clusters: 5

References

[1] The Network Analysis Package. http://igraph.org/redirect.html/. Accessed: 2014-10-23.