Algorithm Design And Analysis
716181 Auckland University of Technology

Hierarchical Agglomerative Clustering to Extract Communities from Social Network Data

09 November, 2014

Zong Yan Li 1300312
Designed and implemented the software and converted this report into TeX.

Yupan Wang 1122001
Performed a review of the literature and investigated an appropriate algorithm, analysing the mathematics.

Jared Kwok 1382534
Designed the .gml import code and created the Powerpoint presentation.

Jaimes Booth 1305390
Implemented the .gml import code, commented the software and created this report framework.

Abstract. Relationships between individuals can not be seen from raw social network data. Can a Java implementation of hierarchical agglomerative clustering be used to extract information from a social network data set to provide knowledge of community relationships? This report describes a Java implementation of an hierarchical agglomerative clustering algorithm. The aim of this project is to carry out an independent study of this algorithmic problem in relation to social network data. The performance of the algorithm is discussed and the project results are presented.

1 Introduction

In today's environment of Social Networking websites, such as Facebook, large amounts of statistical data is collected. However, this data is often ambiguous, irregular and difficult to interpret. Using appropriate methods, hidden structures can be identified and information extracted and analysed from this complex networked data (Yuruk, Mete, Xu, & Schweiger, 2009). This project uses one such approach to analyse data from social network distributions by implementing an hierarchical agglomerative community detection algorithm, highlighting the importance of graph theory in data analysis.

2 Hierarchical Agglomerative Clustering Algorithm to Compute Social Network Communities

Hierarchical Agglomerative Clustering or HAC takes a bottom-up approach, initially treating each node in the network data set as a singleton cluster (or community) and then successively merging (or agglomerating) pairs of clusters until all clusters have been merged into a single cluster that contains all nodes (Manning, Raghavan, & Schütze, 2008).

Merges are determined by choosing the locally optimal choice at each stage with the hope of finding a global optimum. Therefore, it is a greedy algorithm. The results of hierarchical clustering are often presented in a dendrogram. This report, however, presents clustering with coloured edges between nodes, where the same colour represents belonging to a cluster. In this way, the relationship between nodes can easily be visualized in the graph.

A threshold value is provided to specify at above what value nodes will be merged into a cluster.

2.1 Mathematical Analysis

To perform cluster analysis an indicator of "closeness" is needed to decide whether it is possible to merge a specific group of nodes into one cluster. We denote:

$$C = \frac{e}{\frac{n(n-1)}{2}}$$

where *e* is the actual number of edges, *n* is the actual number of nodes and $\frac{n(n-1)}{2}$ is the maximum possible edges in one community. We use the ratio between *e* and $\frac{n(n-1)}{2}$ to calculate the closeness between each sub-community, where a larger *C* indicates a closer relationship. The largest ratio between actual edges and maximum possible edges is 1 as:

$$0 \leq C \leq 1$$

To analyse each cluster, we compare the result of *C* for all communities, and select the largest *C* to merge into the cluster and continue further cluster operations. Therefore, the time complexity is $O(n^3)$.

2.2 Pseudocode Hierarchical Agglomerative Clustering Algorithm

```
Hierachical Cluster:

cluster(thresh)
      initialize all communities
      run community one
      run community two
      calculate cluster coefficient of community one and two
      merge the largest two communities
      remove the original communitie(s)
      return the new merged community

Calculate density:

Identity = e/(n*(n-1))/2
For all nodes in communities
      if node1 is connected to node2
            add existing edges

Density = existing edges / total
```

2.3 Design and Implementation

The HAC algorithm was implemented as a Java software application using the Netbeans Integrated Development Environment. The project contains the following eight classes:

Community.java
- Create initial communities, calculate the density and merge communities by selecting the largest density result in each round of comparisons.

DensityQualityFunction.java
- Calculate the coefficients of communities.

Digraph.java
- Implement Graphic User Interface

HClustering.java
- Cluster communities based on different value of identity (from the formula)

LoadGML.java
- Convert gml dataset into an arrayList data structure in java

MainActivity.java

- Implement a command-line User Interface

Node.java

- Create nodes

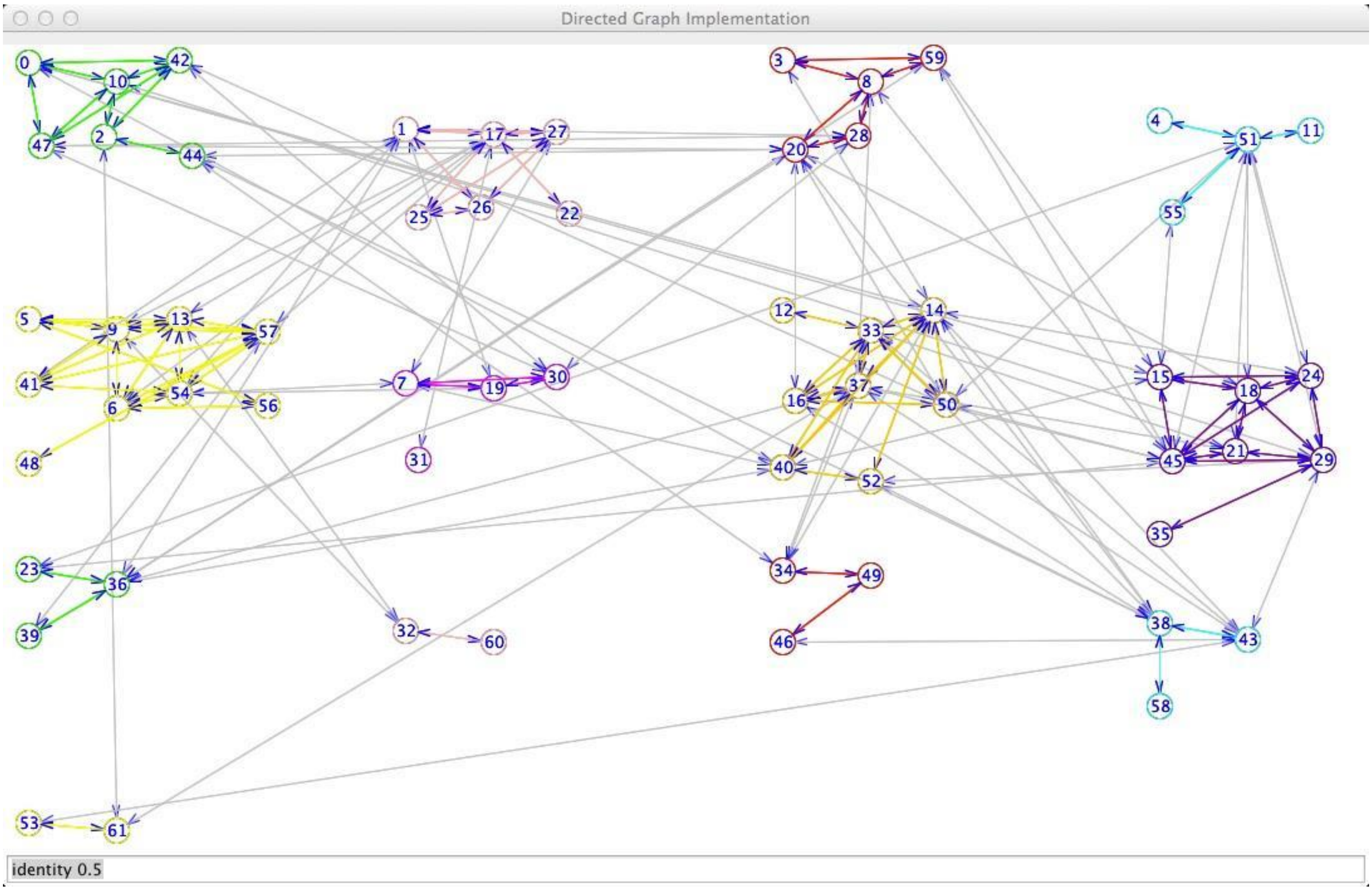
QualityFunction.java

- An Interface for calculating the cluster's coefficients

The data set primarily used for this project was a Graph Modeling Language (.gml) file of a dolphin social network: (Lusseau et al., n.d.) This data set was chosen for the availability of previous clustering results for comparison and for it's New Zealand context.

A second .gml data set, Zachary's Karate club (Zachary, 1977), was also used to further test the algorithm's accuracy against previous analysis.

2.4 Results and Analysis



identity 0.5

Figure 1 *Clustering of Dolphin set using HAC algorithm with threshold of 0.5*

The advantage of this HAC algorithm is that community groups do not need to be defined and these communities can be automatically detected. A disadvantage of this algorithm is that is has a large time complexity of $O(n^3)$. The processing time increases polynomially as the number of input nodes increases. This means the algorithm is too slow for large data sets. (“Hierarchical clustering,” 2014)

The dolphin set is a small set of 62 nodes, small enough to gain insight into the data set by hand. This is too small to see the full advantage of using an automated algorithm. The advantages of the automated algorithm would better be seen on large data sets, where hand processing would be practically impossible.

3 Performance Analysis of Hierarchical Agglomerative Clustering

The following figures provide examples of previous analysis network data sets. These are presented as a comparison of how the implemented HAC algorithm compares with previous analyses.

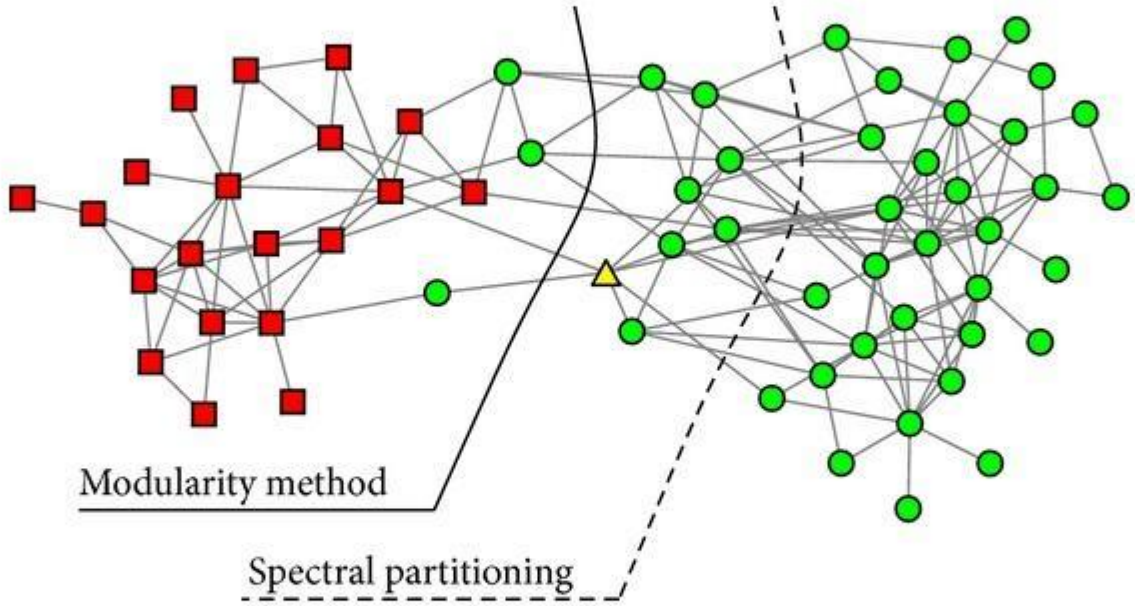
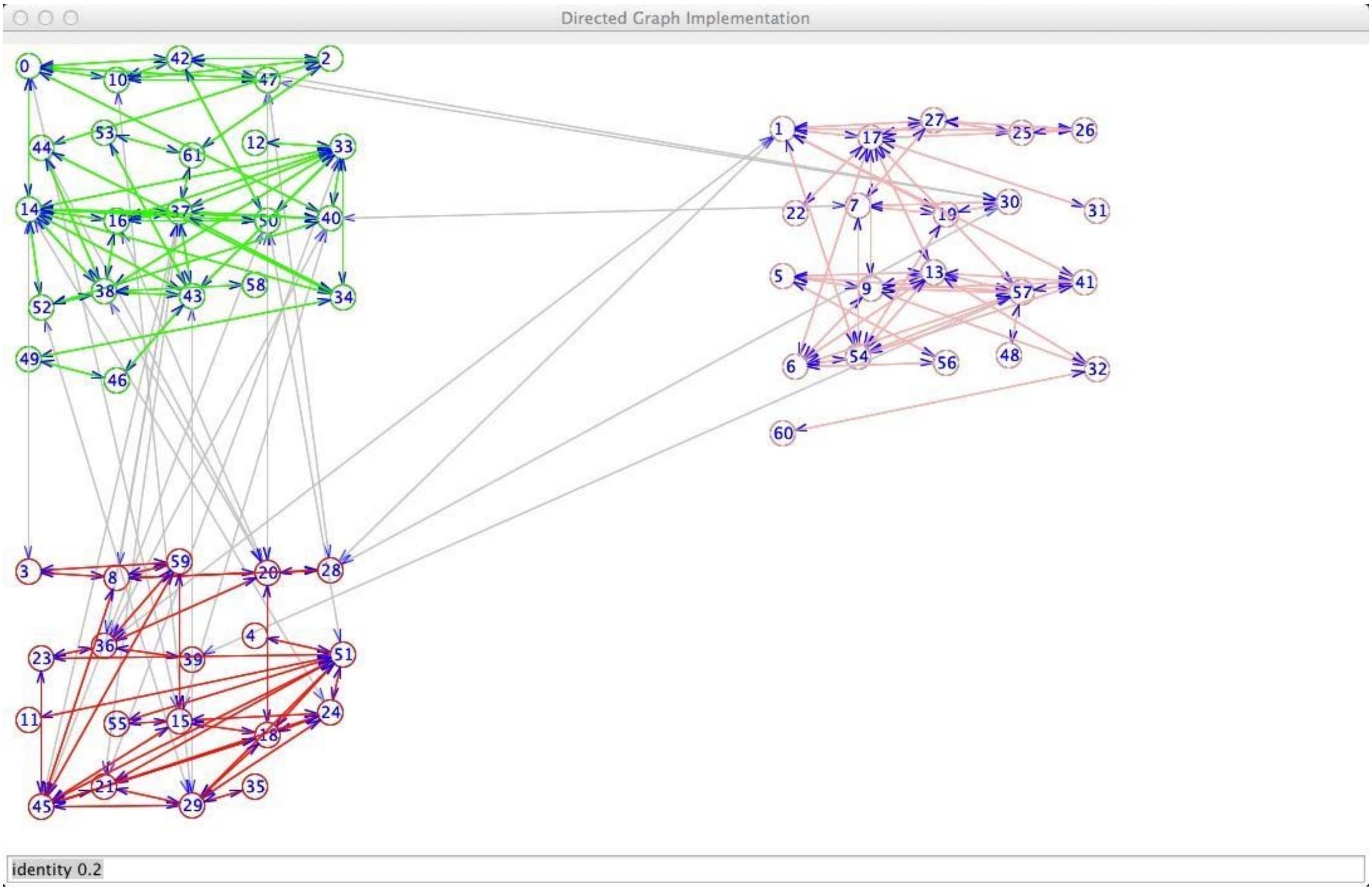


Figure 2 *Clustering of Dolphin set using centrality divided clustering (Wu, Qi, Fuller, & Zhang, 2013)*



identity 0.2

Figure 3 *Clustering of Dolphin set using HAC algorithm with threshold of 0.2*

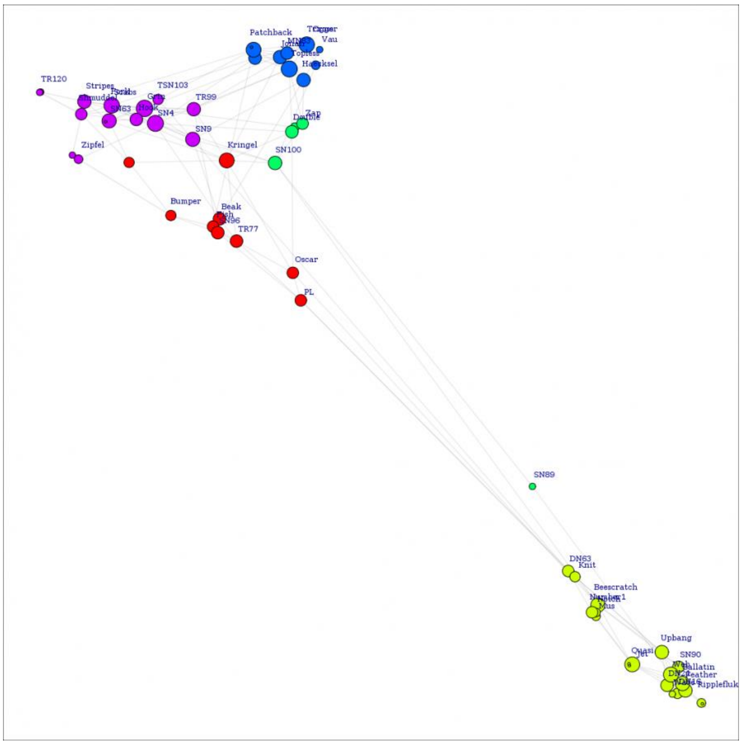
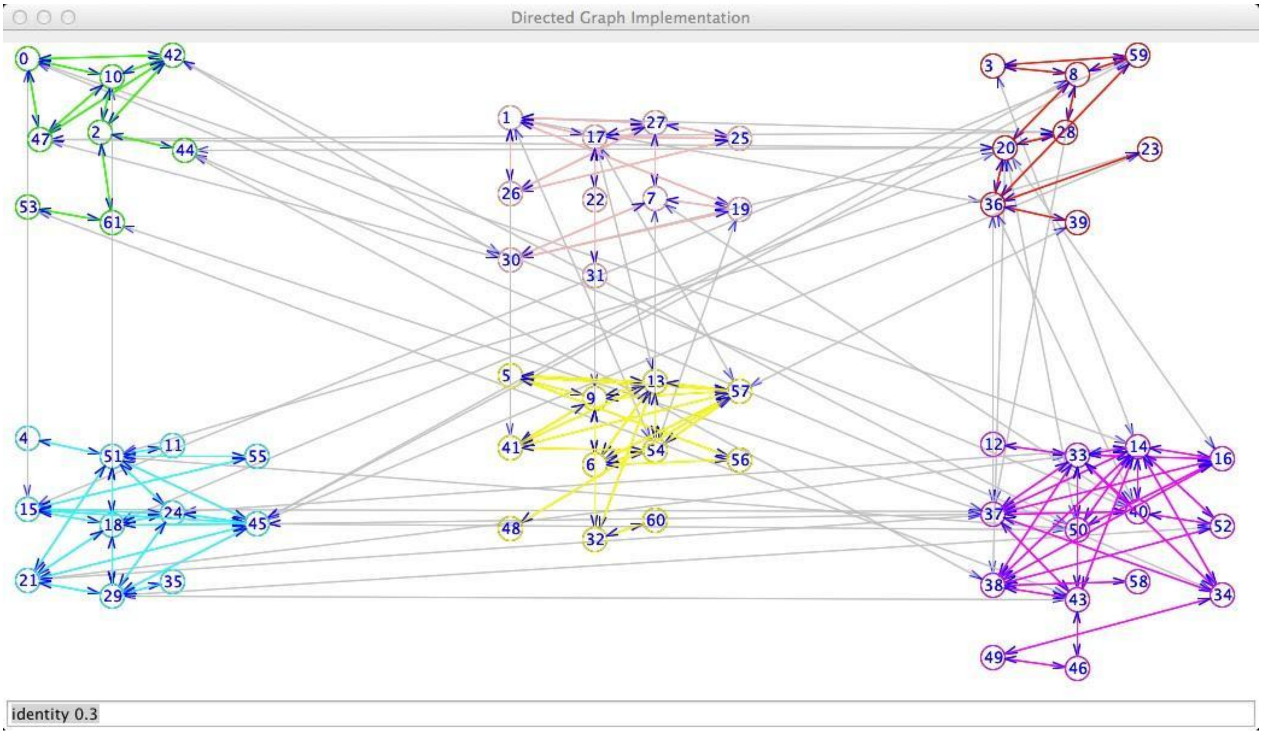


Figure 4 *Clustering of Dolphin data set using multi-level graph clustering (Rotta, 2008)*



Directed Graph Implementation

identity 0.3

Figure 5 *Clustering of Dolphin set using HAC algorithm with threshold of 0.3*

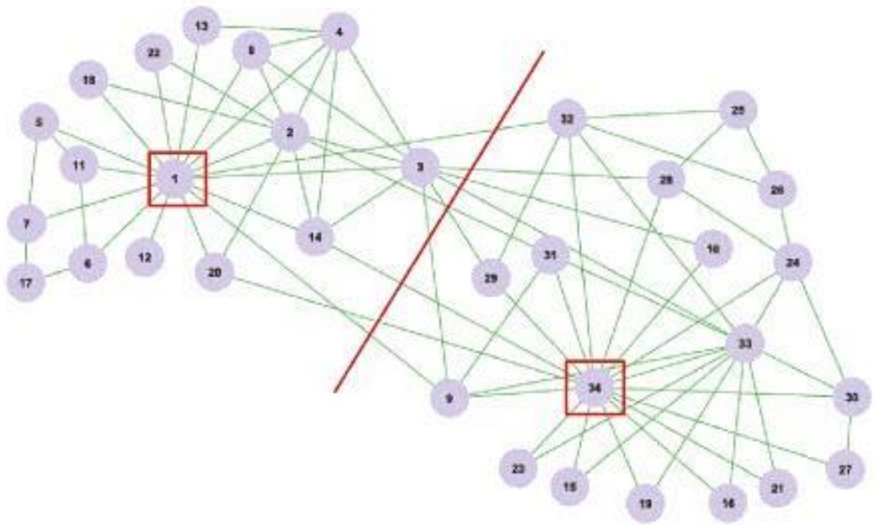
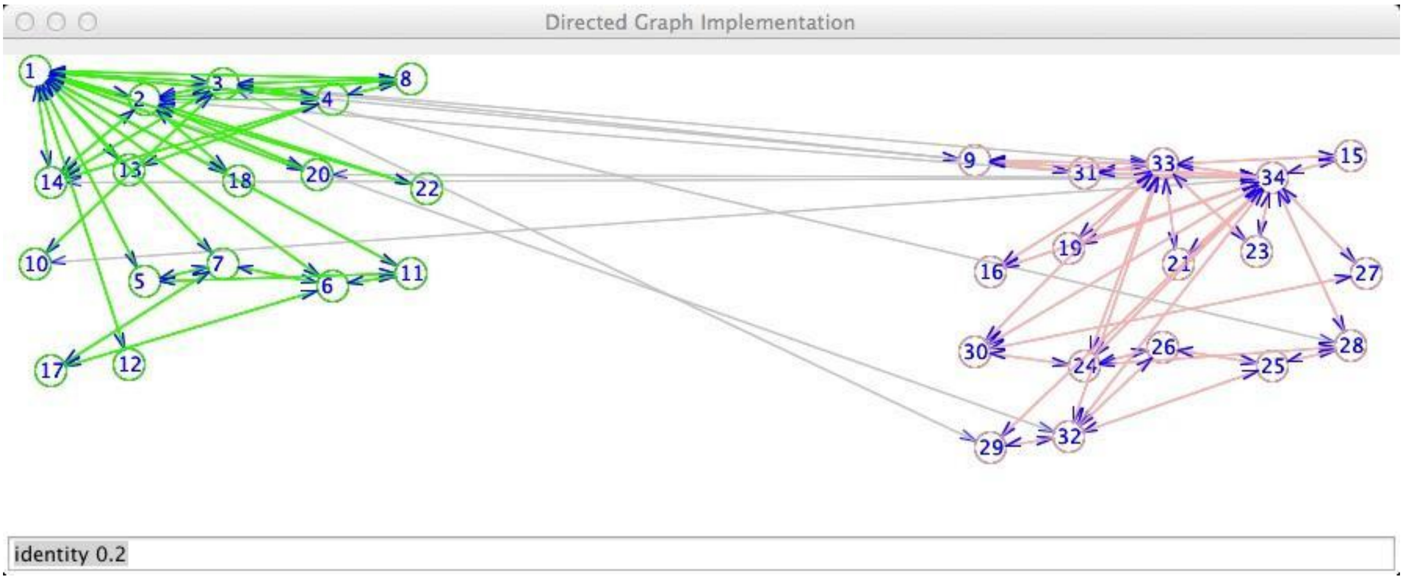


Figure 6 *Clustering of Zachary's Karate club using centrality method (Combinatorial Optimization and Applications - 4th International Conference, COCOA 2010, Kailua-Kona, n.d.)*



identity 0.2

Figure 7 *Clustering of Zachary's Karate club using HAC algorithm with threshold of 0.2*

In this comparison of Zachary's Karate data set, two communities and two influential nodes, 1 and 34 can be seen in both the example graph (**Figure 6)** and the HAC algorithm graph (**Figure 7**)

4 Conclusions

To summarise, an hierarchical agglomerative clustering algorithm was successfully implemented as a Java application. Using this application, communities within a complex network dataset can automatically be detected and visualised. Future implementations could provide an easier interface to select different data sets. A more flexible .gml parser could also be implemented. Further directions for investigation could include measuring the processing time for (large data set) input values to confirm the time complexity experimentally.

References

Behnke, L. (n.d.). lbehnke/hierarchical-clustering-java. Retrieved November 1, 2014, from https://github.com/lbehnke/hierarchical-clustering-java

Blei, D. (2008, February 28). Hierarchical Clustering COS424. Princeton University.

Combinatorial Optimization and Applications - 4th International Conference, COCOA 2010, Kailua-Kona,. (n.d.). Retrieved from http://www.springer.com/computer/theoretical+computer+science/book/978-3-642-17457-5

Davis, T. (2014, March 12). Newman/dolphins sparse matrix. Retrieved November 8, 2014, from http://www.cise.ufl.edu/research/sparse/matrices/Newman/dolphins.html

Dendrogram. (2014, August 22). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Dendrogram&oldid=619966983

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826. doi:10.1073/pnas.122653799

Greenacre, M., & Primicerio, R. (2014). *Multivariate Analysis of Ecological Data*. Fundacion BBVA.

Hierarchical agglomerative clustering. (n.d.). Retrieved November 9, 2014, from http://nlp.stanford.edu/IR-book/html/htmledition/hierarchical-agglomerative-clustering-1.html

Hierarchical clustering. (2014, November 7). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Hierarchical_clustering&oldid=632810346

Hierarchical clustering of networks. (2014, May 31). In *Wikipedia, the free encyclopedia*. Retrieved from http://en.wikipedia.org/w/index.php?title=Hierarchical_clustering_of_networks&oldid=453381996

Lusseau, D., Schneider, K., Boisseau, O. J., Haase, P., Slooten, E., & Dawson, S. M. (2003). The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations - Springer. *Behavioral Ecology and Sociobiology*, *54*(4), 396–405. doi:10.1007/s00265-003-0651-y

Lusseau, D., Schneider, K., Boisseau, O. J., Haase, P., Slooten, E., & Dawson, S. M. (n.d.). Dolphin Social Network Data. Retrieved November 8, 2014, from http://www-personal.umich.edu/~mejn/netdata/dolphins.zip

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press.

Newman, M. (2013, April 19). Network data. Retrieved August 26, 2014, from http://www-personal.umich.edu/~mejn/netdata/

Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. *Physical Review E*, *69*(6). doi:10.1103/PhysRevE.69.066133

Rotta, R. (2008). Multi-Level Graph Clustering Graph: dolphins. Retrieved November 9, 2014, from http://studiy.tu-cottbus.de/~clustering/benchmark_graphs:collection:dolphins

Wu, Q., Qi, X., Fuller, E., & Zhang, C.-Q. (2013). “Follow the Leader”: A Centrality Guided Clustering and Its Application to Social Network Analysis. *The Scientific World Journal*, *2013*. doi:10.1155/2013/368568

Yuruk, N., Mete, M., Xu, X., & Schweiger, T. A. J. (2009). AHSCAN: Agglomerative Hierarchical Structural Clustering Algorithm for Networks. In *Social Network Analysis and Mining, 2009. ASONAM ’09. International Conference on Advances in* (pp. 72–77). doi:10.1109/ASONAM.2009.74

Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, (33), 452–473.