Distributed Graph-Based Clustering for Network Intrusion Detection

Corbin McNeill ◦ Department of Mathematics and Computer Science ◦ Wheaton College, IL
Enyue Lu ◦ Department of Computer Science ◦ Salisbury University, MD
Matthias Gobbert ◦ Department of Mathematics and Statistics ◦ University of Maryland Baltimore County, MD

Abstract

In order to process large volumes of network traffic data and quickly detect intrusions, we use parallel computation frameworks for Network Intrusion Detection Systems (NIDS). Additionally, we model network data as graphs to highlight the interconnected nature of network traffic and apply unsupervised graph-based clustering to flag anomalies as network intrusions. We particularly examine the effectiveness of barycentric clustering on graph network traffic models for intrusion detection. The clustering algorithms have been implemented in Hadoop MapReduce for the purpose of rapid intrusion detection. Furthermore, k-nearest neighbor graphs (kNNGs) are used to optimize the clustering process. We find that across various data sets, unsupervised graph based clustering is able to exceed a 92% intrusion detection accuracy and that kNNGs can effectively optimize the network traffic graphs for larger data sets.

Data and Graph Creation

Network Data

All algorithms were tested using the KDD 1999 Cup's Test Data Set [2]. The KDD 1999 data set lists network connection records and associated connection characteristics. In order to remove many DDOS attacks, intrusions which never made a successful login were removed. All test data sets are random subsets of this filtered data set.

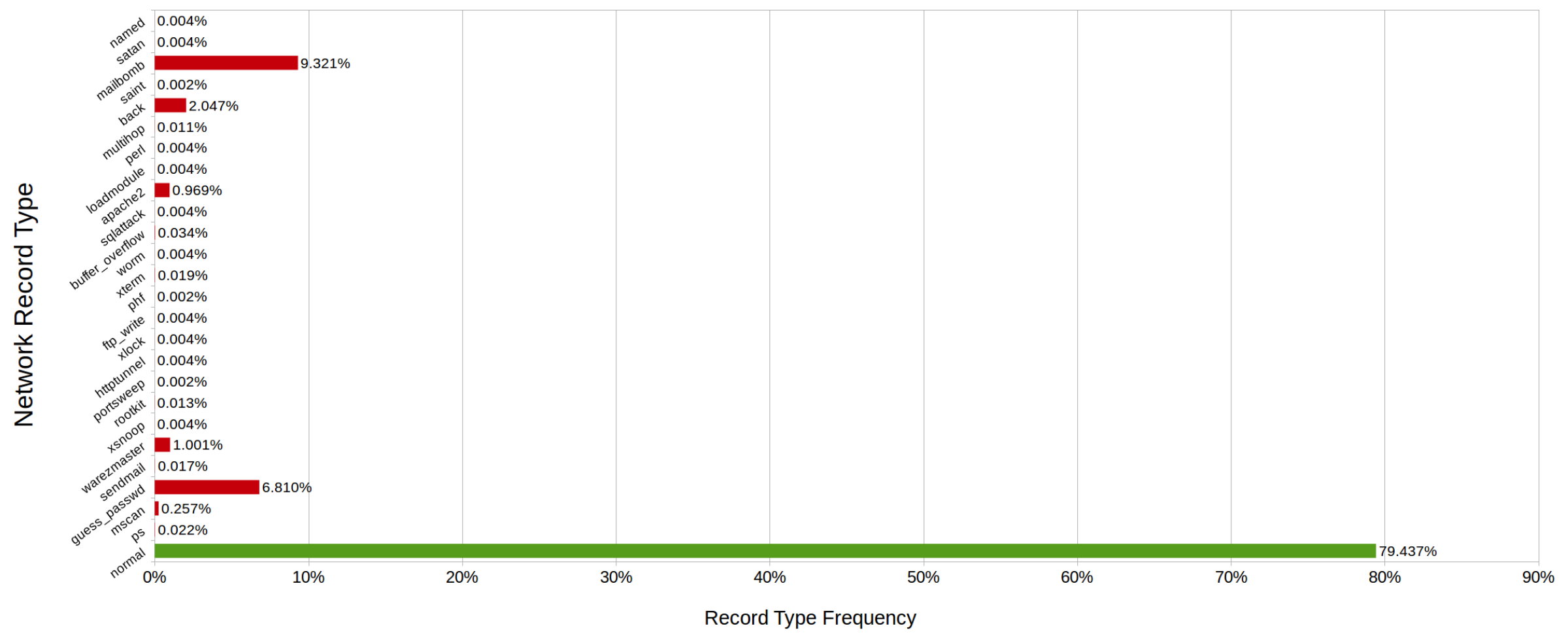


Figure 1 shows the makeup of the filtered data set.

Graph Creation

After the data was filtered, graphs were used to model the data for clustering. In the graph, each network connection is represented by a vertex. Weighted edges between vertices represent the similarity between the two records which the edge connects. Edge weights are calculated by:

$$w_{i,j} = \frac{1}{\exp\left(\left\|\vec{v}_i - \vec{v}_j\right\|^2\right)}$$

where $\vec{v}_x$ is a data vector associated with the x^{th} network record. Each dimension of the data vector represents an attribute from the original data set.

The task of graph creation can be easily distributed using a single MapReduce job. The *mapper* is responsible for creating vertex pairs, and the *reducer* calculates the edge weight for each pair.

Graph Clustering & Intrusion Classification

Clustering Algorithm

Once network traffic data had been modeled in graph format, the graph must be clustered into partitions of similar connections. In order for this clustering to be performed in a timely matter, distributed graph clustering algorithms can be used. Any clustering algorithm capable of clustering undirected weighted graphs could be used for the clustering stage. This versatility is one advantage of graph-based network intrusion detection.

Barycentric Clustering – Hadoop MapReduce

Graphs of network records used in these experiments were clustered using Barycentric Clustering implemented in Apache Hadoop's MapReduce framework.

Barycentric clustering is a clustering algorithm which begins by assigning random positions to each graph vertex and proceeds by iteratively adjusting each vertex's position based on the weights of all adjacent edges. Once vertex positions begin to reach equilibrium, edges of longer than the average length are cut. This concludes a single iteration. Multiple iterations are often performed [1].

Barycentric Clustering requires $O(E)$ time to run, i.e. time proportional to the number of edges [1].

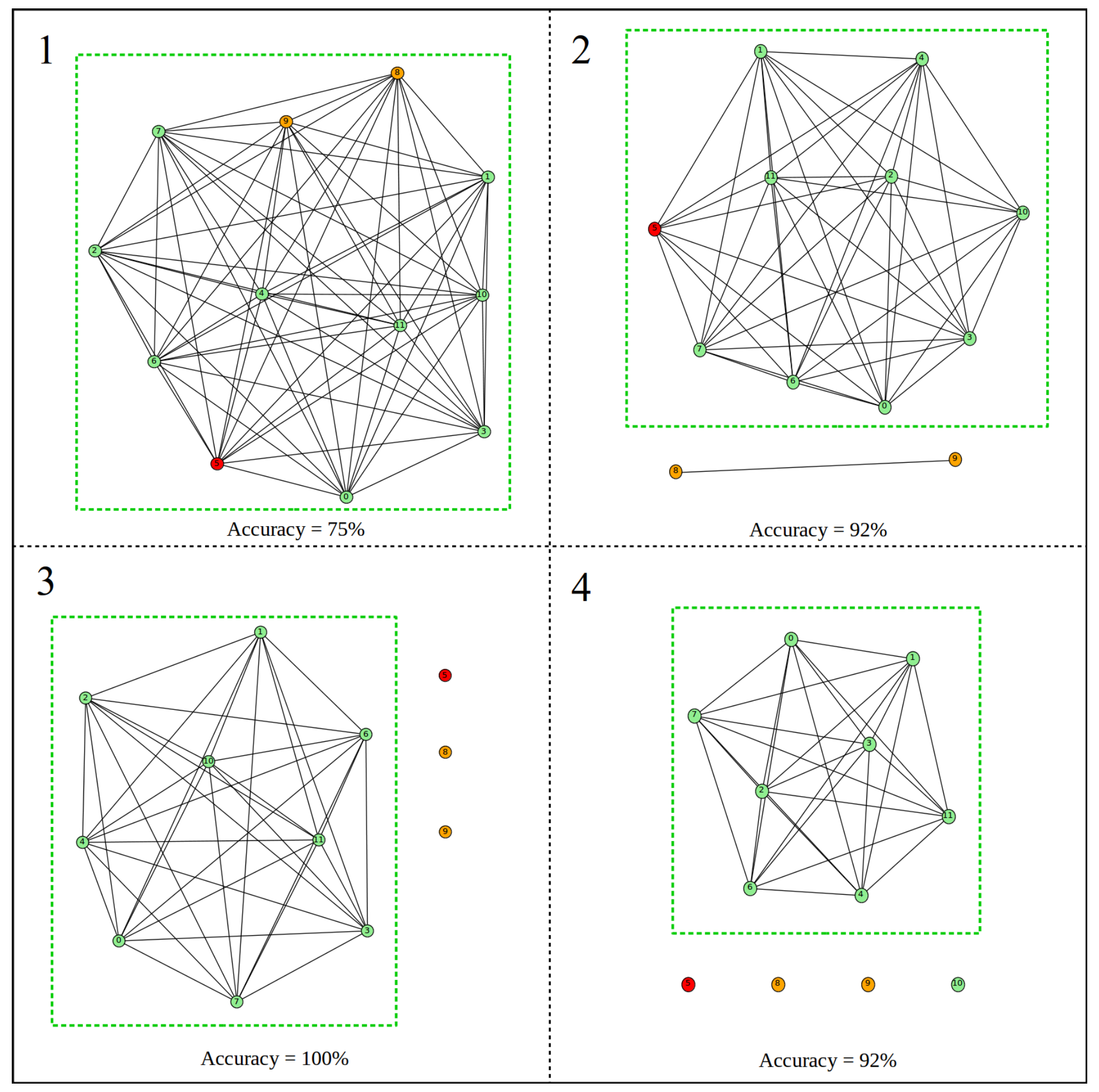
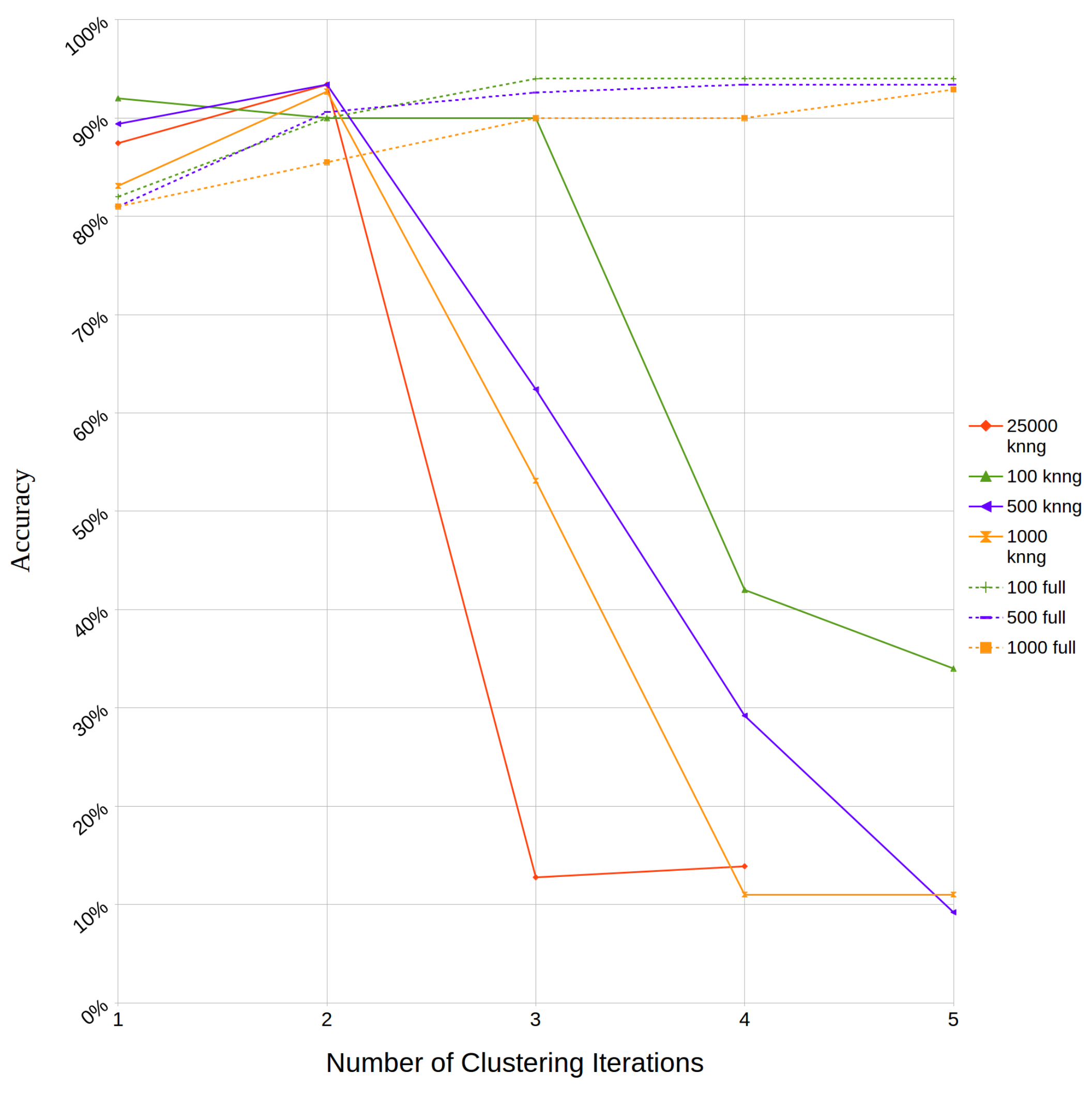


Figure 2 depicts 4 subsequent iterations of the barycentric clustering and intrusion classification process. The data set contains 9 normal records (green), 2 mailbomb intrusions (orange), and 1 rootkit intrusion (red).



Data Set Size	100	500	1,000	25,000
kNNG Accuracy @ 2nd Iteration	90%	93.4%	92.7%	93.4%
Full Graph Accuracy @ 5th Iteration	94%	93.4%	92.9%	N/A

Figure 3 shows detection accuracy at various iterations across data sets. *k*=50

Intrusion Classification

After a sufficient number of barycentric clustering iterations, various connected components will begin to form. An optimal strategy appeared to be classifying every network record vertex within the largest connected component to be *non-intrusive*, and classifying all other network record vertices as *intrusive*.

Detection Accuracy

The effectiveness of graph-based clustering and intrusion classification can be evaluated by via detection accuracy. Detection accuracy is calculated as:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP is the number of *true positives*, FN is the number *false negatives*, etc. In general terms, accuracy is the percent of properly classified records.

Acknowledgement: This work is funded by the National Science Foundation under the Research Experience for Undergraduates Program CCF-1460900 grant.

K-Nearest Neighbor Graphs

Because barycentric clustering (as well as many other clustering algorithms) requires $O(E)$ time, any method of reducing the number of graph edges while maintaining intrusion detection rates would serve as an effective optimization.

Higher weighted edges have a significantly higher effect on the clustered output of barycentric clustering. For this reason, we consider the usage of k-nearest neighbor graphs (kNNGs) instead of fully complete graphs. KNNGs are composed only of edges that are among the k^{th} highest weighted edges adjacent to any vertex for some constant k. Throughout all experiments, a k value of 50 was used.

Figure 3 shows that accuracy is maintained by switching from full graphs to kNNGs, and **Figure 4** shows the extent to which kNNGs reduce the number of edges.

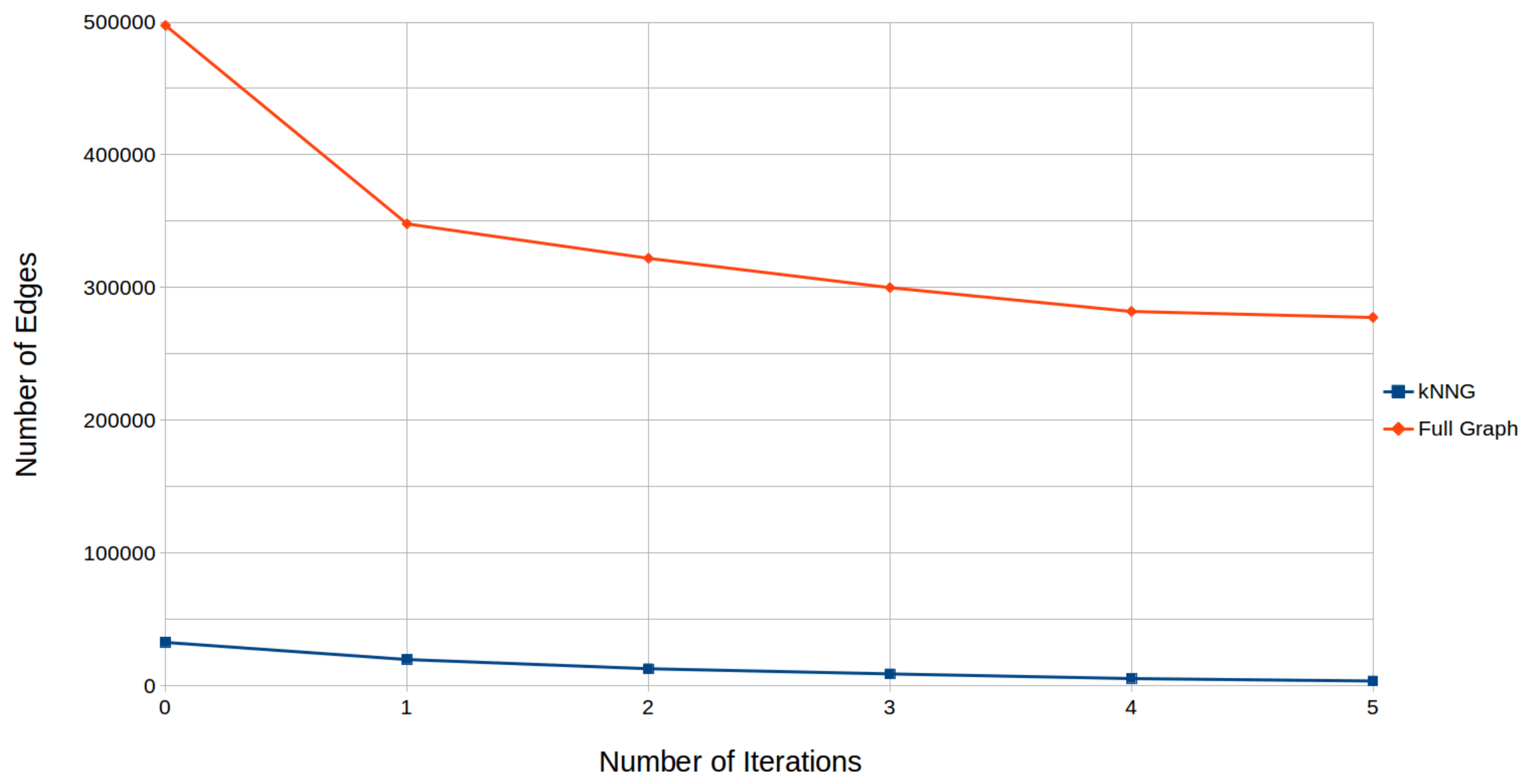


Figure 4 shows the number of edges after various iterations of barycentric clustering for both full graphs and kNNGs. The data used contained 1,000 network records. *k* = 50

Results & Future Work

We find that graph-based clustering algorithms serve as an effective method for unsupervised network intrusion detection with detection accuracy consistently above 92%. Additionally graph creation and graph clustering can be distributed via Hadoop MapReduce and optimized with kNNGs in order to quickly detect network intrusions. KNNG optimizations are very effective for large data sets.

Future work includes investigating the accuracy and efficiency for the intrusion detection through:

1) The usage of other clustering algorithms.
2) The usage of other edge weight formulas.
3) The usage of other *k* values for kNNGs.

References

[1] J.D. Cohen, "Barycentric Graph Clustering," 2008; http://www2.computer.org/cms/Computer.org/dl/mags/cs/2009/04/extras/msp2009040029s2.pdf

[2] KDD data set, 1999; http://kdd.ics.uci.edu/databases/-kddcup99/kddcup99.html