Anomaly Detection

Mike Tsamis and Maria Taranov

Mentor: Dr. James Abello

Anomaly Detection

- Network anomalies typically refer to circumstances when network traffic deviates from normal network behavior.
- Anomaly detection involves analyzing and comparing datasets in order to assess network patterns which are deemed “normal” or “abnormal”.

Graph Visualization

- Networks are often represented by very large graphs that can consist of millions of edges and nodes, so graph visualization software is often used to view, analyze and interact with graphs.
- Interactions include reducing depth, labeling nodes, and clustering the overall graph.
- Examples of Graph Visualization software: Cytoscape and ASK-GraphView.

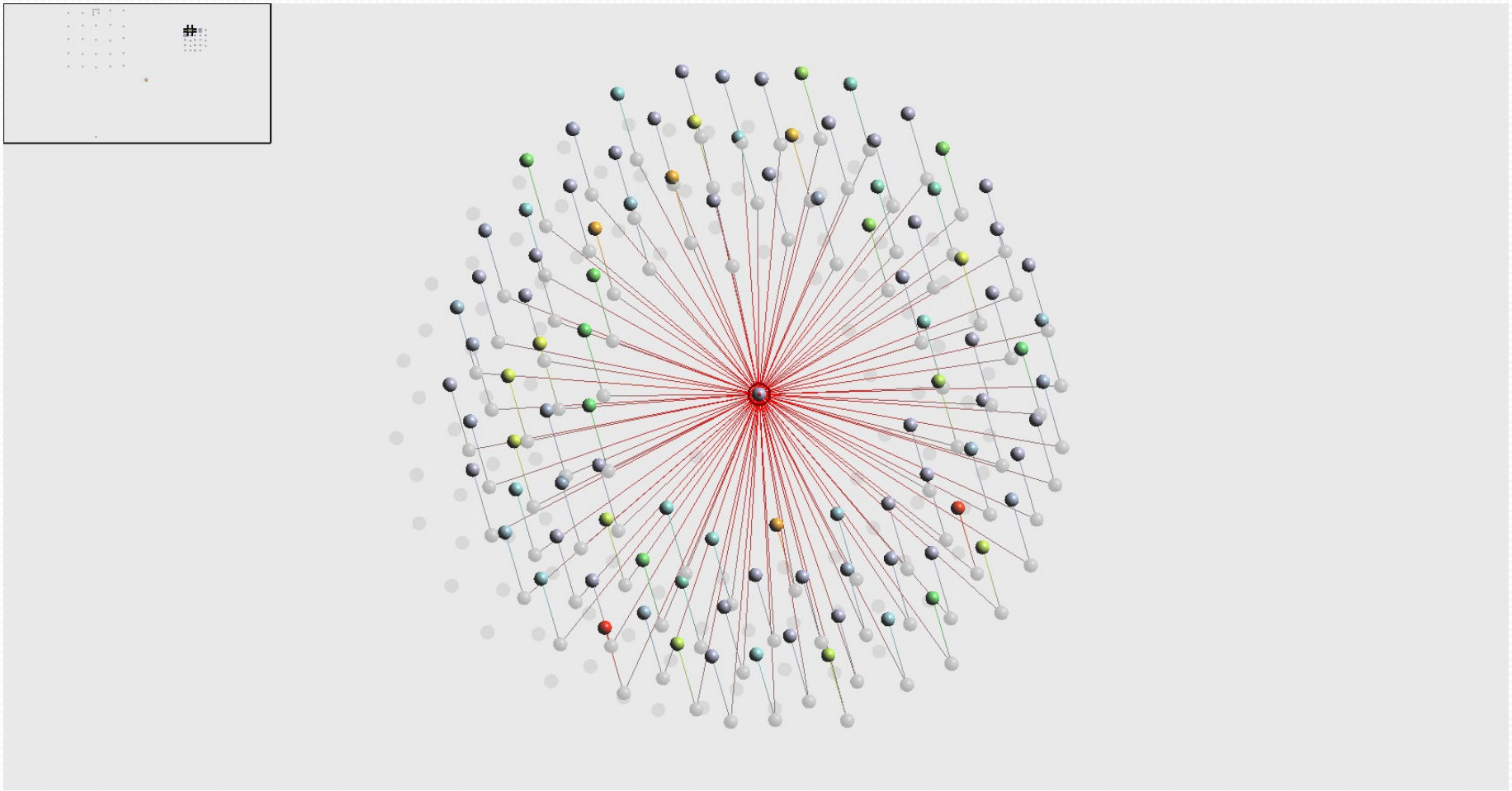
Our Project

- This project addresses the problem of finding persistent patterns in evolving networks. A central question is the characterization of patterns that can be used as the basis to detect anomalous activities in time-evolving networks.
- The datasets we worked with were from Twitter.

Obtaining Twitter Data

- We used an API capable of retrieving Twitter data from users whose tweet contains a specific search term. The following data was retrieved:
  - The user who tweeted the message
  - The Tweet message
  - A URL to the user's profile picture
  - A timestamp for the tweet
  - The user's ID number
  - The language the tweet is written in
  - The user IDs for any users linked within the Tweet via the "@" symbol
  - A URL to the Tweet
- We collected over 150,000 Tweets which contained #Libya, #Egypt, #Yemen, #Tunisia, and #Syria which are all countries that are currently undergoing revolutions.

Example:



First steps

- We started by becoming familiar with our graph visualizations software, ASK-Graphview. We spent the first week visually analyzing graphs of retweets used by previous researchers. We noticed clear patterns. For example, some power users were connected to many nodes and some strongly connected users shared multiple neighbors. Although these patterns were interesting, the graphs were so large that it was impossible to only analyze them visually. Mathematical analysis was needed.

Finding Important Nodes

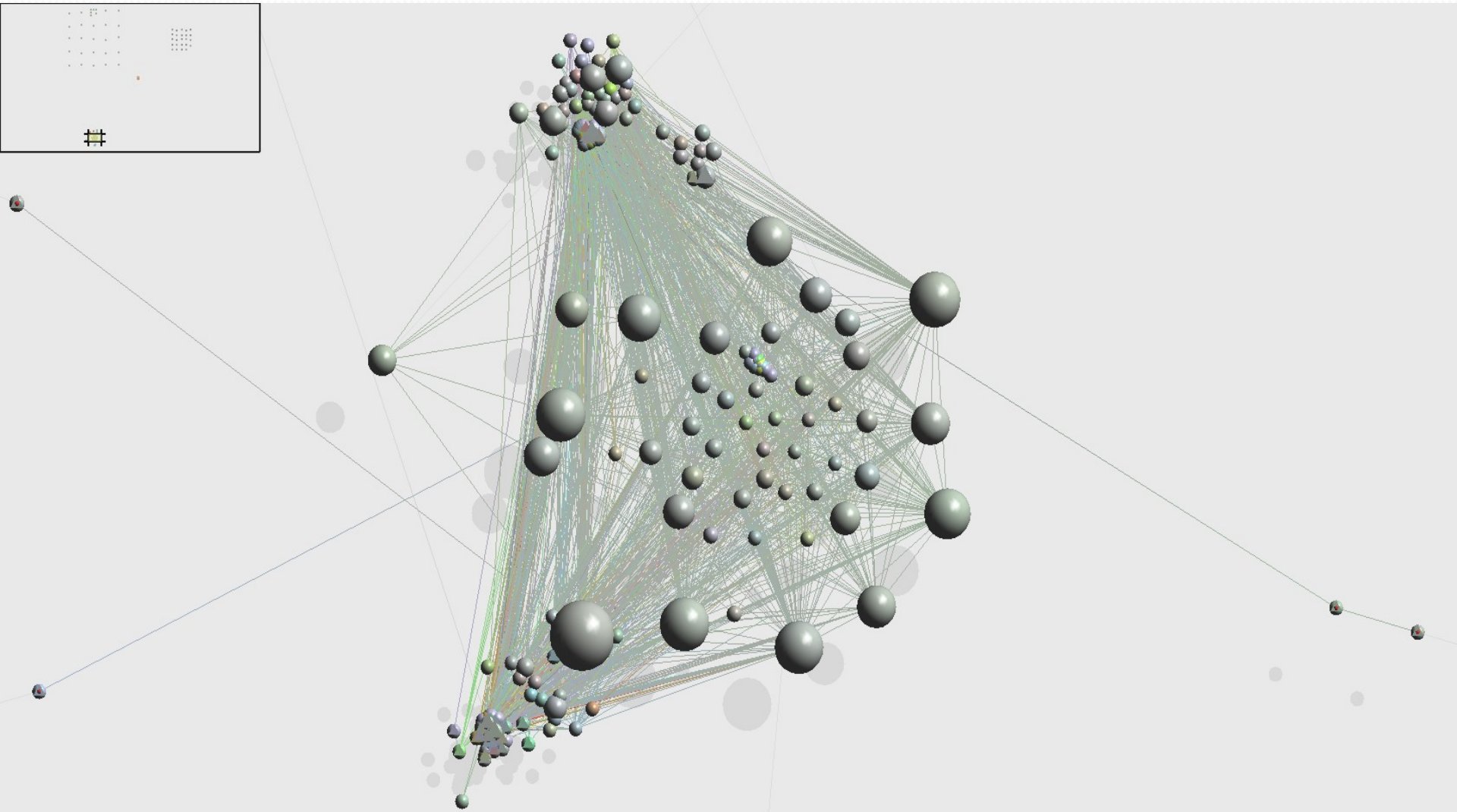
- We wanted to find the most important clusters and the most important nodes within a cluster.
- Denseness (the number of edges divided by the number of possible edges) helps us find strongly connected clusters.
- Membership into a cluster (the ratio of neighbors in a cluster to neighbors outside of the cluster) helps us find nodes that strongly belong to a cluster. Nodes with very high membership into a cluster tend to be similar.

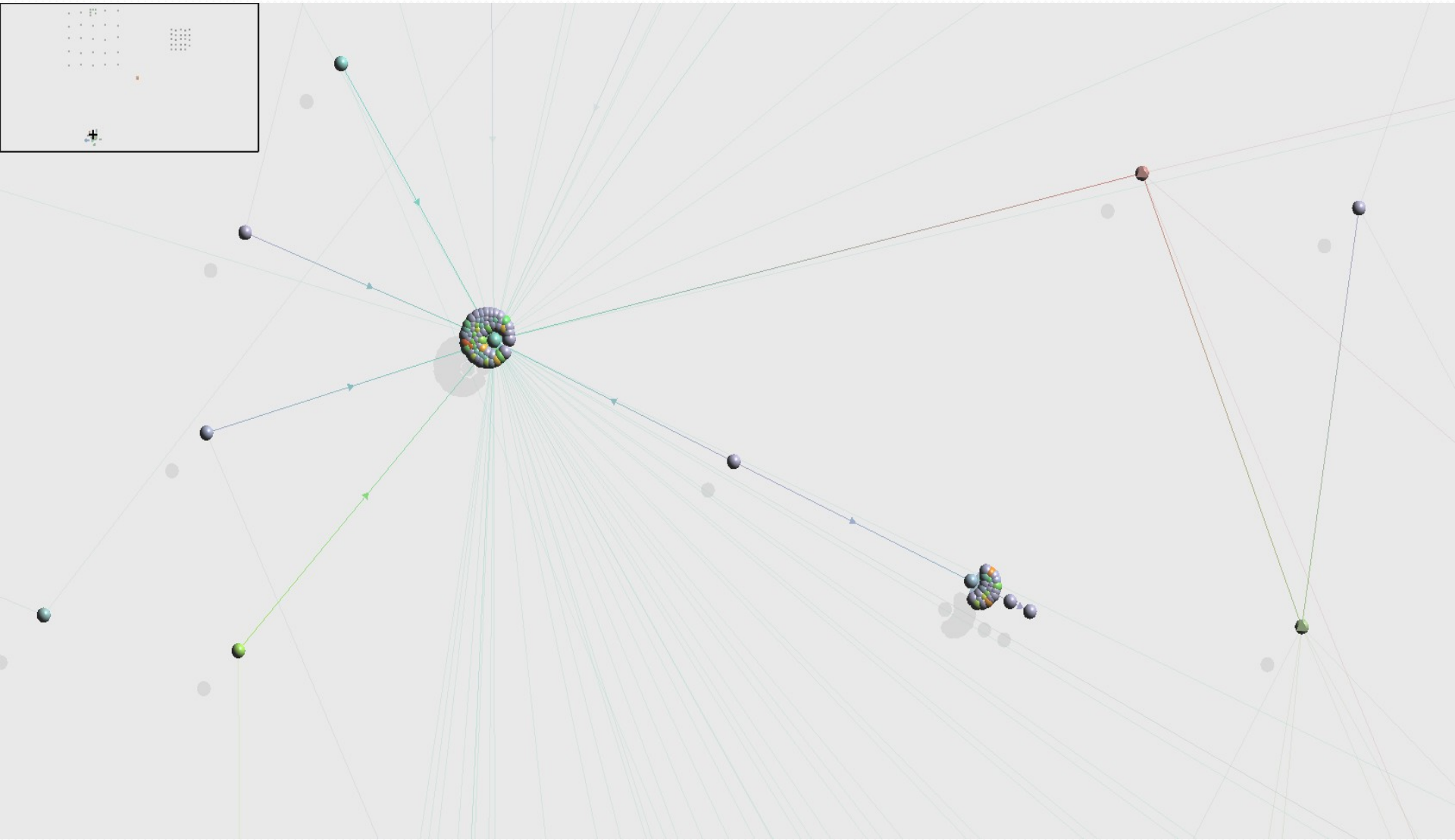
Using Vectors

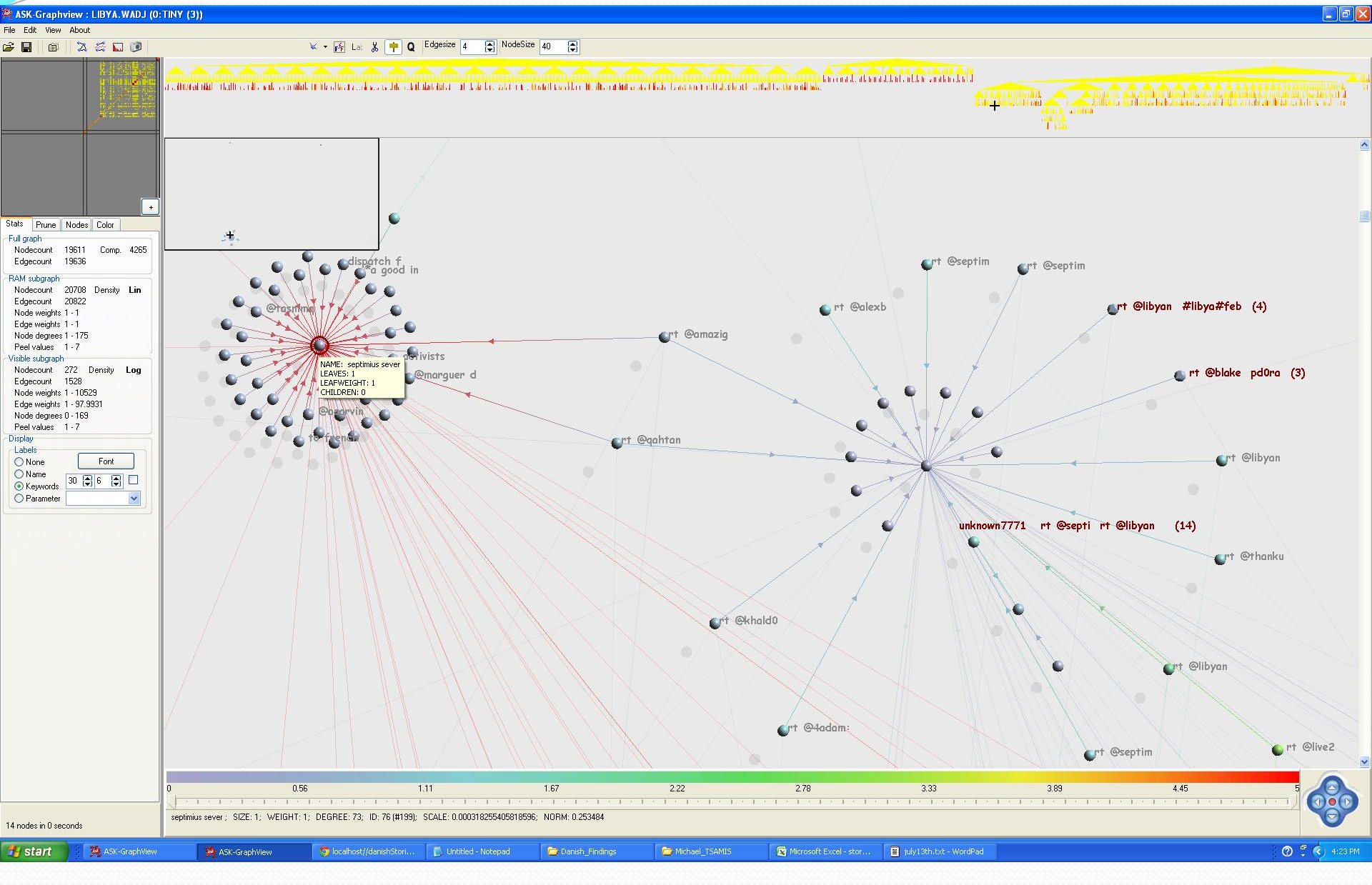
- Each node has membership into each cluster. The clusters can be ordered, and a vector can be formed for each node's membership into the clusters, where the first element is membership into the first cluster and so on.
- The norm (the "length" of the vector) gives us a sense of the average membership. Nodes with high norm are either very interconnected or very strongly connected to a particular clusters. These nodes tend to be important.
- The dot product between two nodes indicates how strongly connected the two nodes are to the same clusters.

Implementation

- Our first step was to process the data and put it into an appropriate format. Then, we obtained a program[4] to cluster the graph. We wrote programs that used the clusters obtained to calculate the membership vectors, their norms and the dot product for each set of connected nodes. This dot product could be used as an edge weight in addition to the timestamp converted to epoch time, which we used.
- The final step was creating an adjacency list file with our calculated edge weights and a labels file that included the calculated clusters and membership norm. The additional attributes allow us to view the graph differently and to view only particular parts of the graph.


(C) ASK-Graphview. James Abello and Frank van Ham




ASK-Graphview : LIBYA.WADJ (0:TINY (3))
File Edit View About
Edgesize 4
NodeSize 40
Stats Prune Nodes Color
Full graph
Nodecount 19611 Comp. 4265
Edgecount 19636
RAM subgraph
Nodecount 20708 Density Lin
Edgecount 20822
Node weights 1 - 1
Edge weights 1 - 1
Node degrees 1 - 175
Peel values 1 - 7
Visible subgraph
Nodecount 272 Density Log
Edgecount 1528
Node weights 1 - 10529
Edge weights 1 - 97.9931
Node degrees 0 - 169
Peel values 1 - 7
Display
Labels
None
Name
Keywords
Parameter
Font
NAME: septimius sever
LEAVES: 1
LEAFWEIGHT: 1
CHILDREN: 0
dispatch f
a good in
@rashima
@marguer d
rt @amazig
rt @alexb
rt @septim
rt @septim
rt @libyan #libya#feb (4)
rt @blake pd0ra (3)
rt @qahtan
rt @libyan
unknown7771 rt @septi rt @libyan (14)
rt @thanku
rt @khald0
rt @libyan
rt @4adam:
rt @septim
rt @live2
0 0.56 1.11 1.67 2.22 2.78 3.33 3.89 4.45 5
septimius sever ; SIZE: 1; WEIGHT: 1; DEGREE: 73; ID: 76 (#199); SCALE: 0.000318255405818596; NORM: 0.253484
14 nodes in 0 seconds
start
ASK-GraphView
ASK-GraphView
localhost//danishStori...
Untitled - Notepad
Danish_Findings
Michael_TSAMIS
Microsoft Excel - stor...
july13th.txt - WordPad
4:23 PM

References

1. M. Thottan and C. Ji, Anomaly Detection in IP Networks, IEEE Transactions on Signal Processing, Vol. 51, No. 8, August 2003.
2. J. Abello, F. van Ham and N. Krishnan, ASK-GraphView: A Large Scale Graph Visualization System, IEEE Transactions on Visualization and Computer Graphics, Vol. 12, No. 5, September/ October 2006.
3. A. Clauset, M.E.J. Newman and C. Moore, "Finding community structure in very large networks." Phys. Rev. E 70, 066111(2004).
4. The clustering program was given to us by Tina Eliassi-Rad and used an algorithm developed by Clauset, Newman and Moore[3].
5. All graphs were created using ASK-GraphView.

Questions?