TTS Assignment 4 Report

Matric no: s0792198

PageRank
To calculate the PageRank of an email address (node) we require: the damping factor, number of iterations, inlinks, outlink count (for each node with an outlink), and the total number of nodes in the graph.

To store the inlinks for any node (email address) in the graph, I created a dictionary and stored each receiver email address as the *key* and a list of email addresses (those who send an email to the receiver) as the *value*. Prior to storing these inlinks in the dictionary from the graph.txt file, I ensured to ignore any email links where sender == receiver. This ensured no self links were included in the graph and that also any nodes that did not receive or send an email to anyone other than themselves (islands) were excluded from the graph.

To store the outlink count for any node, I first repeat the (inverted) process for calculating inlinks, creating a dictionary and ensuring that the *key* is the sender and the *value* is the list of receivers. I then use this dictionary to create an outlink count dictionary where the *key* is the sender and the *value* is the number of that sender's outlinks.

The number of iterations is given as 10 and the damping factor is 0.8. I calculated the total number of nodes N by storing all senders and receivers into a dictionary and simply extracting the (distinct) keys and finding the length of this. For statistics sake I also found the dangling nodes by extracting the receivers who are not senders.

After processing; removing the islands and selflinks, we have 86029 total nodes, 19488 senders, 77012 receivers and 66541 dangling nodes. It was interesting to see a large number of self emails, a large number of dangling nodes and malformed addresses.

To compute the PageRank: the PageRank for each node is initialised to 100% / total nodes. For the remaining iterations, the PageRank for each node is calculated by the following formula as given in class: $PR(x) = \frac{1-\lambda}{N} + \lambda \sum_{y \to x} \frac{PR(y)}{out(y)}$ [3]
The PageRank for each node was then stored and the old initial PageRank updated (for use in the next iteration).I did not normalize the PageRank after each iteration as I aimed to keep the results inline with the unnormalised sanity checks. The top 10 highest PageRanks were stored in the file pr.txt

Hubs and Authorities
For HITS, we can re-use the data stores we used for PageRank; the inlinks, outlinks and total nodes. I created a dictionary the Hubs scores and another dictionary for the Authority scores with the *key* corresponding to the node (email address) and the *value* corresponding to either the Hubs or Authority score.

To compute the HITS algorithm, each node has it's Hubs and Authority score initialised to 1. For the remaining iterations, we run the Authority update rule, then run the Hub

update rule [3]. The Hub and Authority scores are then normalised and stored in the dictionaries.

I saved the 10 highest Hubs scores to a file called hubs.txt and the 10 highest Authority scores to auth.txt.

Visualise Key Connections

Looking at the list of enron employees, it becomes quite clear that PageRank is quite accurate in finding key influential individuals. Comparing the list of enron employees with important titles such as Vice President, Manager and CEO against my list of PageRanks shows that these individuals have high ranks. This is likely due to the fact that their emails pass on a portion of their high PageRank to each other. Thus important employees who email each other tend to cluster around the same high ranking. Most employees that I checked against my rankings tended to be in the top 10%.

Likewise, Hubs and Authorities showed for example that many of these top employees such as Vice Presidents had high Authority scores.

Graph Analysis

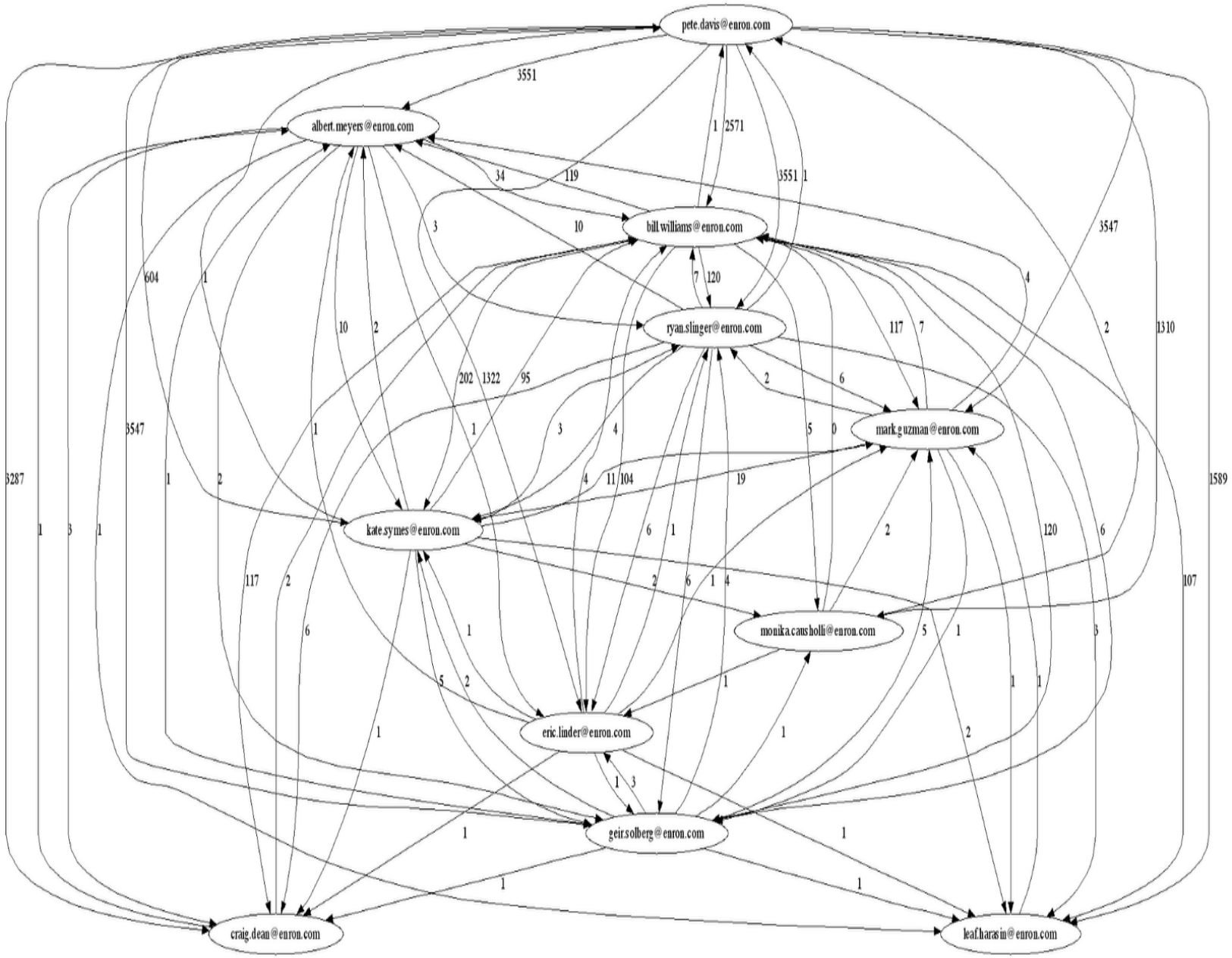
I noticed that *"pete.davis@enron.com"* had sent the largest number of emails. Looking at his outgoing emails showed that the recipients were almost always a list of around 10 similar people. These repeat emails seemed interesting to me; is Pete Davis relaying news to a list of key people? Are these recipients part of some group? Is Pete Davis a proxy for relaying information to a group of people (and thus bypassing direct messaging)?

Without looking at the email content, I decided to use the message with id (0018fb24de98cc0f07d9e865d760e44f) for my graph representation. I used Pete Davis along with all the recipients as nodes in the graphs and added labels indicating the number of emails sent from one node to another.

This graph showed that emails among these recipients was limited and sometimes non-existent. However they all received a large number of emails from Pete Davis. Very few emails are sent back to Pete Davis. To me this represents an group of people worth investigating.

References

[1] http://en.wikipedia.org/wiki/PageRank
[2] http://www.inf.ed.ac.uk/teaching/courses/tts/pdf/web-2x2.pdf
[3] http://en.wikipedia.org/wiki/HITS_algorithm


pete.davis@enron.com
albert.meyers@enron.com
bill.williams@enron.com
ryan.slinger@enron.com
mark.guzman@enron.com
kate.symes@enron.com
monika.causholli@enron.com
eric.linder@enron.com
geir.solberg@enron.com
craig.dean@enron.com
leaf.harasin@enron.com