Konstantinos Ilias
DS210 Project Write-Up

Dataset/Project Description
This project uses data from email communication of employees of Enron, a company that filed for bankruptcy in 2001 because it committed accounting fraud.

I found the data here:https://snap.stanford.edu/data/email-Enron.html

I am using the SNAP Enron dataset(email-Enron (1).txt) which is a dataset of edges where each node is an email address. Although the original email data is directional meaning one user sends a message to another, the SNAP Enron dataset treats the email communication as undirected, meaning an edge exists between two nodes if at least one email was exchanged regardless of its direction.

The SNAP database provides numeric node IDs with no direct email mapping. However, another dataset was provided (enron_mail_20150507.tar.gz), which contains over 500,000 emails organized in employee folders within a maildir directory, where each folder (e.g., lay-k, skilling-j) represents an Enron employee's mailbox. Sender and receiver email addresses are provided as well.

I used python to create another csv file that maps the numerical nodes to the email addresses and employee folders. With python I produced email_to_node.csv, which mapps NodeID to Email to Folder. Emails and folders can be different as only employees have folders but the dataset has a lot of non Enron emails. Therefore, these non Enron emails are saved in folders of Enron employees.

What does the code do, how to run it?
The program performs three types of centrality analysis (Degree, Closeness, Betweenness) and finds clusters to identify representative nodes within each cluster. Given that degree centrality is the least computationally heavy to implement, I calculated closeness and betweenness centrality only on the top 1000 nodes with the highest degree centrality. The code prints the top 10 nodes by: 1) Degree Centrality 2)Closeness Centrality 3)Betweenness Centrality and also prints cluster leaders (top node by degree in each cluster). It finally generates the following plots:
1)degree_histogram.png
2)closeness_vs_degree.png
3)betweenness_histogram.png

- Degree Centrality tells us how many direct connections a node has, indicating how active or visible an individual is in the network.

- Closeness Centrality measures how close a node is to all others via shortest paths, identifying those who can quickly communicate with everyone else.

- Betweenness Centrality captures how often a node lies on the shortest paths between other nodes, highlighting individuals who serve as key connectors or information brokers.

To run the code one needs these two datasets: email-Enron (1).txt, email_to_node.csv, these four rust modules:

- main.rs: Uses all functions to print the top 10 nodes given different centrality measures
- graph.rs: Reads the files, conducts the mapping and computes centrality measures
- cluster.rs: Divides nodes into clusters with the help of BFS
- plot.rs: Generates plots using the plotters crate
- tests: Contains tests for the important functions

and an environment that supports Rust and cargo. Using the cargo run –release command the program takes around 5 seconds to generate the following output.

What is the output?

The code produces rankings of the most important individuals in the Enron email network based on three centrality metrics. Nodes with the highest degree centrality (e.g., bdbinford@aol.com, .marisha@enron.com) had the most direct email connections, meaning they were highly active in sending and receiving emails. Nodes with high closeness centrality could reach others quickly in the network, indicating their ability to efficiently disseminate information. Those with high betweenness centrality often served as connections between different groups, playing a role in information flow and inter-group communication. Additionally, the network was divided into clusters of connected individuals, and the most connected node in each cluster was identified as its leader, revealing the most central figure within each communication cluster.

These four executives got convicted of fraud: **Jeffrey Skilling** – Former CEO, **Andrew Fastow** – Former CFO, **Kenneth Lay** – Former Chairman and CEO and **Richard Causey** – Former Chief Accounting Officer.

Based on the output of the code: **Jeffrey Skilling** appears in **Closeness Centrality** and **Cluster 4** and **Kenneth Lay** appears in **Betweenness Centrality**.

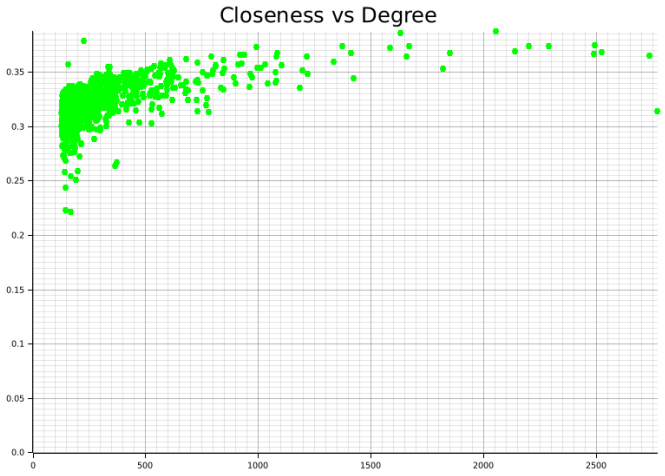
Output when using cargo run - -release

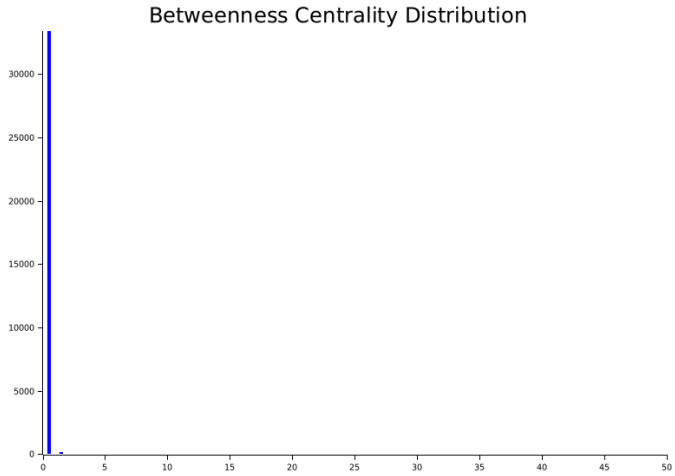
```
$ cargo run --release
    Finished `release` profile [optimized] target(s) in 0.18s
     Running `target/release/project`

 Top 10 by Degree Centrality:
 1. Node 5038 (bdbinford@aol.com) [horton-s]: 2766 connections
 2. Node 273 (.marisha@enron.com) [wolfe-j]: 2734 connections
 3. Node 458 (09najjarna@bp.com) [weldon-c]: 2522 connections
 4. Node 140 (.cody@enron.com) [lucci-p]: 2490 connections
 5. Node 1028 (7u9k73h@msn.com) [wolfe-j]: 2488 connections
 6. Node 195 (.gerald@enron.com) [nemec-g]: 2286 connections
 7. Node 370 (09acomnes@enron.com) [steffes-j]: 2198 connections
 8. Node 1139 (a..hueter@enron.com) [shapiro-r]: 2136 connections
 9. Node 136 (.cindy@enron.com) [ward-k]: 2052 connections
10. Node 566 (151@artic.net) [harris-s]: 1848 connections

 Top 10 by Closeness Centrality:
 1. Node 26576 (k.naughton@pecorp.com) [shively-h]: 0.22190
 2. Node 9137 (chris.cockrell@enron.com) [shankman-j]: 0.22345
 3. Node 20764 (hrobertson@cloughcapital.com) [arnold-j]: 0.24413
 4. Node 16201 (epao@mba2002.hbs.edu) [maggi-m]: 0.25097
 5. Node 16202 (eparson@miamiair.com) [love-p]: 0.25458
 6. Node 9129 (chris.benham@enron.com) [fischer-m]: 0.25871
 7. Node 8344 (ccampbell@kslaw.comjkeffer) [mann-k]: 0.25909
 8. Node 5022 (bcrena@pkns.com) [sanders-r]: 0.26418
 9. Node 5069 (bdrinkwater@mcdonaldcarano.com) [steffes-j]: 0.26728
10. Node 8556 (celemi@excelcomm.com) [skilling-j]: 0.26907

 Top 10 by Betweenness Centrality (top 1000 nodes only):
 1. Node 140 (.cody@enron.com) [lucci-p]: 1.00000
 2. Node 5038 (bdbinford@aol.com) [horton-s]: 0.83232
 3. Node 1139 (a..hueter@enron.com) [shapiro-r]: 0.64237
 4. Node 195 (.gerald@enron.com) [nemec-g]: 0.60960
 5. Node 566 (151@artic.net) [harris-s]: 0.60922
 6. Node 273 (.marisha@enron.com) [wolfe-j]: 0.58823
 7. Node 458 (09najjarna@bp.com) [weldon-c]: 0.58023
 8. Node 136 (.cindy@enron.com) [ward-k]: 0.57691
 9. Node 292 (.ned@enron.com) [lay-k]: 0.55936
10. Node 588 (1800flowers.209594876@s2u2.com) [campbell-l]: 0.53313

 Cluster Leaders by Degree:
 Cluster 1 (33696 nodes) → Node 5038 (bdbinford@aol.com) [horton-s], Degree: 2766
 Cluster 2 (6 nodes) → Node 33463 (members@premierinvestornetwork.com) [carson-m], Degree: 10
 Cluster 3 (12 nodes) → Node 27851 (khara@avistaenergy.com) [scholtes-d], Degree: 22
 Cluster 4 (4 nodes) → Node 35866 (myates@continuum-corp.com) [skilling-j], Degree: 6
 Cluster 5 (2 nodes) → Node 19420 (groceryworⓇprocessrequest.com) [cuilla-m], Degree: 2
 Cluster 6 (2 nodes) → Node 33192 (mcunningham@isda.org) [haedicke-m], Degree: 2
 Cluster 7 (2 nodes) → Node 36473 (newssubs@iso-ne.com) [thomas-p], Degree: 2
 Cluster 8 (6 nodes) → Node 31354 (maco.fowlkes@enron.com) [parks-j], Degree: 10
 Cluster 9 (4 nodes) → Node 34873 (mmacias@spl-inc.com) [griffith-j], Degree: 6
 Cluster 10 (2 nodes) → Node 31100 (lvalentin@poten.com) [smith-m], Degree: 2
```

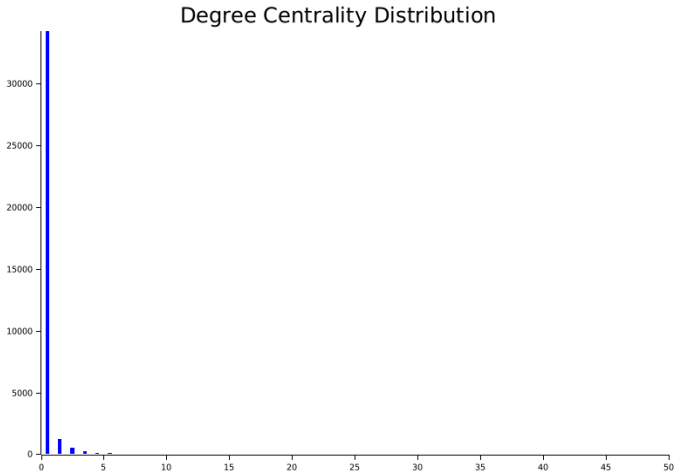
Output when using cargo test

```
running 4 tests
test cluster::tests::test_find_clusters ... ok
test graph::tests::test_compute_betweenness ... ok
test graph::tests::test_compute_closeness ... ok
test graph::tests::test_compute_degree ... ok

test result: ok. 4 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

Graphs


Closeness vs Degree
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0.0
0
500
1000
1500
2000
2500


Betweenness Centrality Distribution
30000
25000
20000
15000
10000
5000
0
0
5
10
15
20
25
30
35
40
45
50


Degree Centrality Distribution
30000
25000
20000
15000
10000
5000
0
0
5
10
15
20
25
30
35
40
45
50