Bitcoin Trust Analysis

CS 225 Final Project

THE DATASET

- Bitcoin OTC trust weighted signed network
- Rankings of trust (-10 to 10) of different traders in this network
- Cleaned up using Pandas library in Jupyter Notebooks
    - While developing the algorithms we were able to visualize, compare, and verify the results using Jupyter notebooks alongside unit testing

PROJECT GOALS

1

PAGERANK

Ranks traders by trust and finds least/most trustworthy traders

2

BFS TRAVERSAL

Traverses the graph and finds path between a source/target

3

BETWEENNESS CENTRALITY

Finds the trade with the most transactions

GRAPH CLASS

- The Graph class is an **Adjacency List** based graph which uses the CS225 Edge class (modified to fit our requirements).
- The **Adjacency List** keeps a track of every User present in the data set and links them to all the incoming/outgoing transactions the User is a part of.

BFS TRAVERSAL

- Our BFS traversal is a quite basic algorithm tailored to our program
- We paired it with a **findPath( )** which accurately finds the path between a source and target vertex

PAGERANK ALGORITHM

- Used by Google to optimize their search engine; measures importance of web pages
- Allows us to rank traders and find out who is the most trustworthy or least trustworthy
- Initializes to a default value → checks outlinks/inlinks of vertices → adds values to those with more transactions
- With each iteration we get an accurate picture of the overall ranking of traders

BETWEENNESS CENTRALITY

- Finds the trader through which the most transactions pass through
- Calculates a centrality number based on a specific vertex's dependencies
- Iterates through each source/target pair and calculates distance from the source using a map of predecessors/processes number of shortest paths to each of the nodes from the source

TESTING PROCESS

- Using the NetworkX Python library, we ran the algorithms we coded in a Jupyter Notebook.
- We imported the Catch testing module by modifying the templated CS225 makefile to fit our use case.
- We then wrote assertions to make sure that our functions gave us the required outputs as well as by testing with **main( )**.

OUTCOMES & IMPLICATIONS

- Successfully met project goals!
- Learned how to implement such complex algorithms
- **Future Development:** potentially publish this online with a GUI
- **Real World Implications:** has capacity to be used in research/investigation on such networks
    - ie.) Bitcoin transactions are often used in highly illegal activities and such a program that ranks traders/finds those through which most transactions pass through can be of great assistance

THANK YOU


NEWS