CS6890: Fraud Analytics
Assignment 2
Implementation of TrustRank using Pregel Framework for Bad Node Identification

Manikanta Vallepu - AI20BTECH11014
Jarupula Saikumar - CS21BTECH11023
Vignan Kota - CS21BTECH11029
Sahil Chandra - CS20BTECH11033
Kalala Abhinav - CO21BTECH11007

1. Problem Statement

In a network of nodes, the goal is to **identify bad nodes** using edge weights, which represent the strength of the transaction between a group of nodes. This will be accomplished by creating an efficient technique that **uses patterns in the connections between the nodes** to identify malicious or other "bad" nodes. A **distributed computing framework** should be used to implement the resulting methodology since it should be **scalable to large networks**.

In this project, we use a seed set of bad nodes provided by the oracle to discover bad nodes using TrustRank. Because of this, the final rank will be **lower for excellent nodes and higher for bad nodes**. This is intuitively comparable to **DistrustRank**. We handle the method's scalability by utilising the Pregel framework.

2. Description of the dataset

There are two files in the dataset: "**Payments.csv** and "**bad_sender.csv**." A directed graph with nodes for the sender or receiver and edges for the transactions between them can be created using the values in the former. The significance and power of the transaction between the seller and the buyer are represented by each edge. It should be mentioned that the nodes' IDs are mapped to 0–799 for convenience's sake without sacrificing generality. When two nodes have more than one edge connecting them, the edges are sometimes consolidated by adding up their values. Synthetic linkages are made back to all of the problematic nodes with equal weightage in the event that a node has no outlinks, which could result in leakage.

The IDs of 20 bad nodes are included in the later file, which serves as the study's seed set.

2.1. Statistics of the dataset

- `Payments.csv`: consists of a directed graph with the sender and the receiver represented by nodes and transactions between them represented by edges.
  - `Sender`: Node of the sender.
  - `Receiver`: Node of the receiver.
    - The IDs are mapped in such a way that it starts from 0, for the sake of convenience without loss of generality.
  - `Amount`: Amount of transaction between the sender and the receiver.
    - There are cases where there exists multiple links between the same two nodes. In those cases, the links are consolidated by summing up the values.
  - Counts:
    - Unique nodes: 799
    - Nodes with no outgoing link: 96

- `bad_sender.csv`: List of bad nodes for using as a seed set.
  - Bad nodes: 20

3. Algorithm Used

3.1. TrustRank

An oracle-provided trusted seed set of pages is incorporated into TrustRank, a **topic-specific version of PageRank**. Using these reliable seed nodes to find further reliable pages in a web graph is the aim of TrustRank. This group of reliable seed nodes serves as the starting point for the process, which **propagates trust scores along the graph edges iteratively**. The TrustRank equation is shown in Equation 1, which takes the edge weights and seed set into account. Algorithm 1 offers a brief pseudo code representation for reference in order to make things clear.

Algorithm 1 TrustRank Algorithm with Pregel Framework

Require: Seed set for initialization (d), Link Weightage (ρ), Iterations (N), Damping factor (β)
Ensure: Nodes with no outlinks are connected back to the seed nodes with equal weightage (to prevent leakage)
while $n \leq N$ **do**
Parallel computation for all nodes u**:**

$R(u) = \sum_{v \in B_u} \rho_v \cdot R(v) \cdot \beta + (1 - \beta) \cdot \frac{1}{d_u}$

Main TrustRank Formula

end while

In this study, we employ TrustRank to identify bad nodes by utilizing a seed set of bad nodes provided by an oracle. Consequently, the resulting rank will be higher for bad nodes and lower for good nodes. This concept can be intuitively understood as **DistrustRank**.

$$R(u) = \sum_{v \in B_u} \rho_v * R(v) * \beta + (1 - \beta) * \frac{1}{d_u} \tag{1}$$

Where:

- u = a node
- B_u = the set of u's backlinks
- ρ_v = weightage of $R(v)$ based on the number of forward links of page v and its corresponding edge value between node v and u
- β = damping factor
- d_u = node u's initialized value at start based on the provided seed set

3.1.1 Handling Leakage

Nodes with no outgoing links tend to accumulate the incoming values and cause leakage. To address this issue, in this work, **those nodes are connected back to all the bad nodes with equal weightage**. This approach ensures that the accumulated values are evenly propagated back to the bad nodes.

3.2. Pregel Framework

To enhance scalability, we utilize a simplified version of Google's distributed-computing framework known as **Pregel**. This framework is designed to efficiently process large directed graphs across a cluster of computers. At its core, Pregel enables each node to operate independently by maintaining its state, receiving messages from its neighbors, and updating its state accordingly. Although the current dataset does not require such a scalable framework, this implementation serves as a demonstration of Pregel's effectiveness. It is important to note that in this implementation, each thread simulates the presence of multiple clusters of computers, with each thread handling the communication of a subset of nodes.

3.3. Hyperparameters

- 'num_threads' : 4
- 'damping_factor' : 0.85
- 'iterations' : 100

4. Results

The code takes a directed graph as input and computes the scores for each node in the graph. The output of the algorithm is a list of **DistrustRank** scores, one for each node in the graph.

Although there is no ground-truth data available for direct comparison, we can evaluate the effectiveness of the algorithm by examining the ranks of the oracle-provided bad nodes, which are expected to be high if the algorithm performs optimally.

- From Figure 1, it is evident that the majority of the ground-truth bad nodes have higher scores. This **indicates that the algorithm has performed as intended**.
- Figure 2 illustrates that the majority of nodes have ranks between 0.00 and 0.01, with only a small number of nodes possessing higher ranks. This distribution suggests that the majority of nodes are likely to be good nodes, with only a select few exhibiting characteristics indicative of distrust.
- Figure 3 illustrates the iteration-wise TrustRank of each node, highlighting the bad nodes. Similar to what was observed in Figure 1, most of the bad nodes exhibit high ranks. However, **it is noteworthy that certain nodes display significantly higher scores than the bad nodes. These nodes are identified as predicted bad nodes, and it is recommended that authorities prioritize investigating these sender or receiver for potential fraudulent activities**.

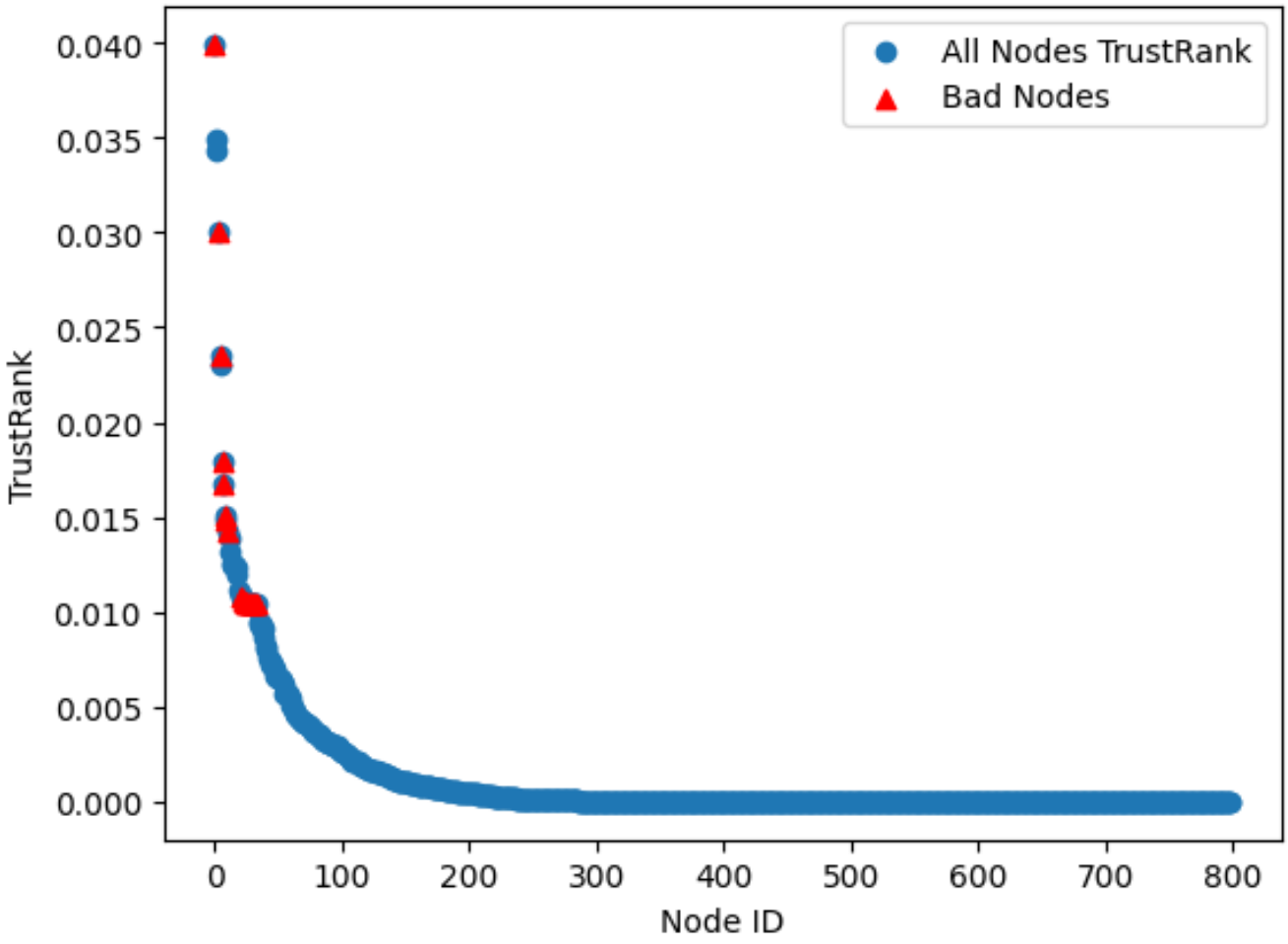

Figure 1. All nodes are arranged in the descending order of their scores and bad nodes are highlighted.

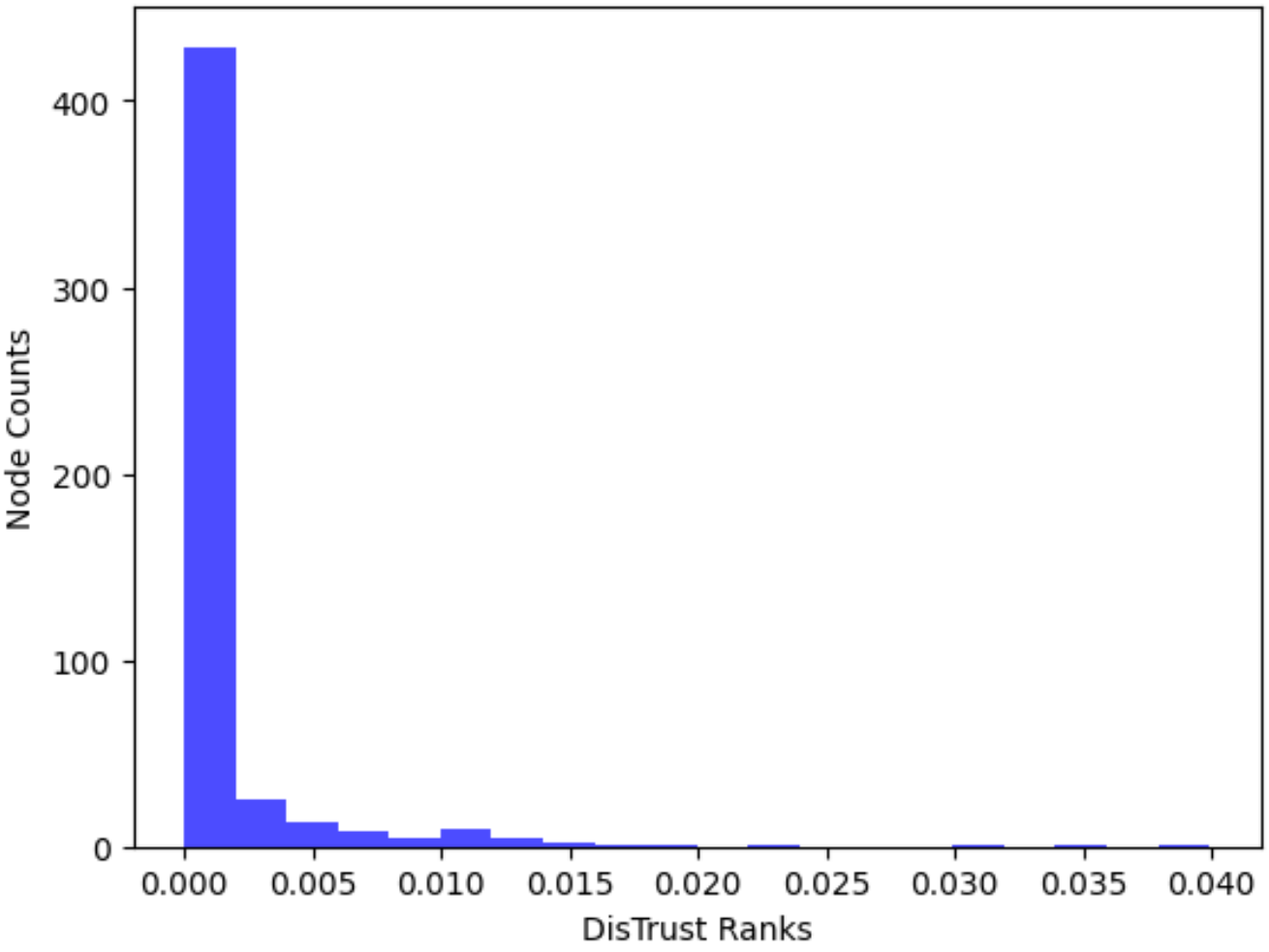

Figure 2. Histogram of the ranks of all the nodes.

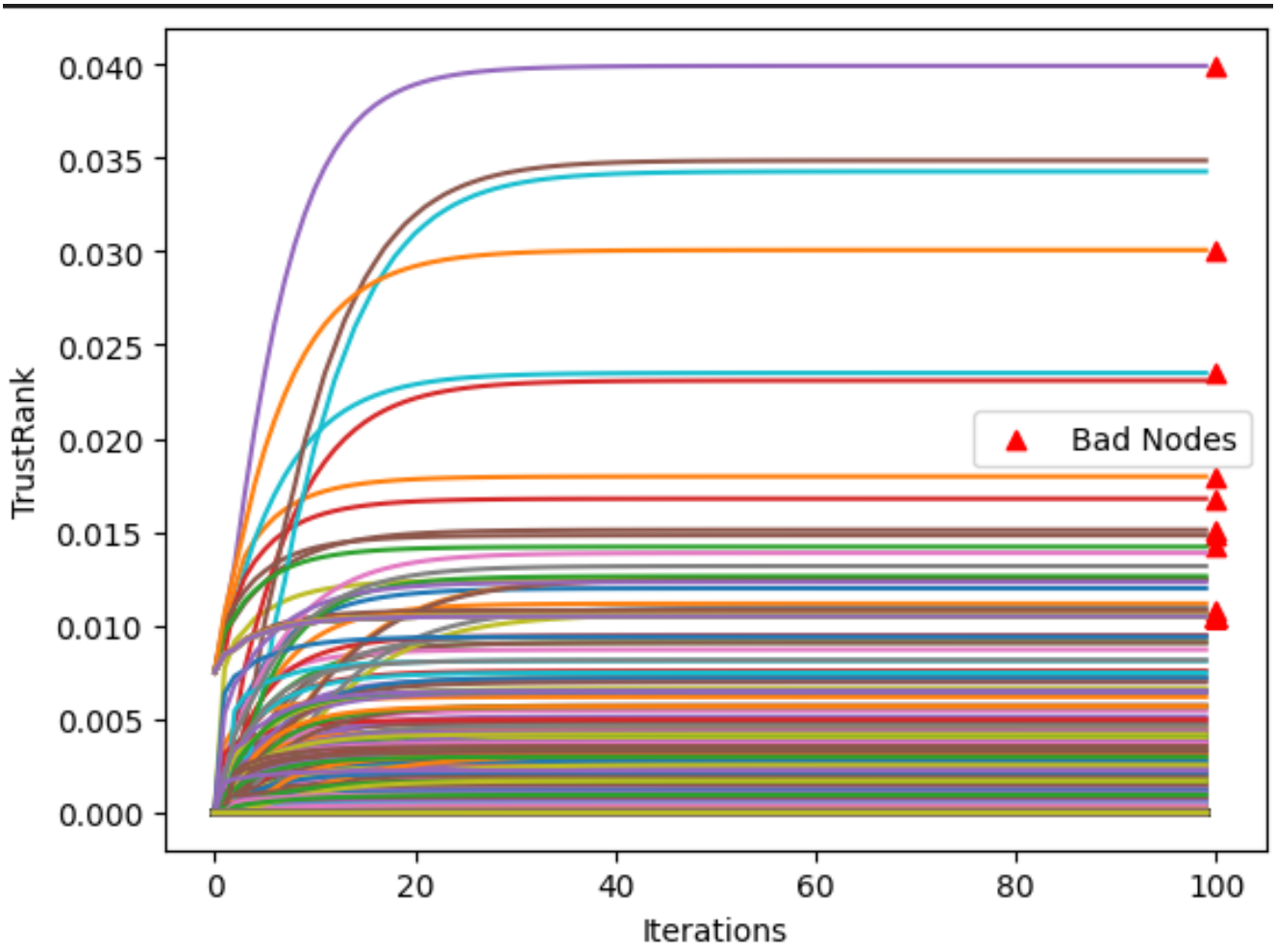

Figure 3. Iteration-wise scores of all the nodes and bad nodes are highlighted.