Trust Rank Algorithm Implementation

Aabisheg T (es21btech11001)
Ranveer Sahu (es21btech11025)
S Jagadeesh (es21btech11026)
Subiksha Gayathiri KK (es21btech11031)
Bhende Adarsh Suresh (cs21btech11008)

May 2, 2024

1 Abstract

Web spam pages employ diverse strategies to attain artificially inflated rankings in search engine results. Though human experts can detect spam, manually evaluating a vast number of pages proves prohibitively costly. Consequently, we employ an iterative method known as the PageRank algorithm to rank web pages. This approach can be extended to assess a page's trustworthiness through the TrustRank algorithm.

2 Problem Statement

Web spam pages use various techniques to achieve higher-than-deserved rankings in a search engine's results.While human experts can identify spam, it is too expensive to manually evaluate a large number of pages. Thus, we use an iterative approach to ranking pages on the World Wide Web called the PageRank algorithm. This approach can further be extended to compute the reliability of a page, using an algorithm called the TrustRank algorithm. This TrustRank methodology will make use of the biased PageRank Algorithm, which assigns a different static score to each node(Sender/Receiver) depending upon how reliable they are

3 Dataset Description

The dataset comprises payment transactions, each record indicating a sender, receiver, and transaction value. An additional file identifies 20 bad nodes,suspected of engaging in undesirable activities. The dataset encompasses 703 unique senders, 371 unique receivers, with 275 entities serving both roles. In total, there are 799 nodes (unique senders and reeivers) in the dataset.There are approx. 1,30,000 invoices in the dataset. Dataset when visualized as a graph, it results into a dense network of directed edge weighted multigraph. These findings suggest a diverse network of financial transactions, with overlapping relationships and potential areas of concern regarding the integrity of certain entities.

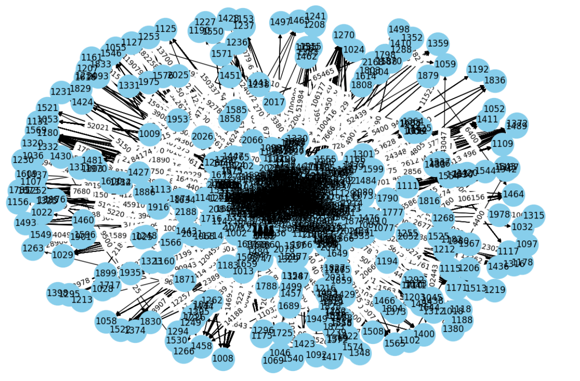


Figure 1: Directed Edge Multigraph of Transactions

4 Algorithm Used

4.1 Introduction to Web Model

The web is represented as a directed graph $G = (V, E)$, where V is the set of pages (vertices) and E is the set of directed links (edges) connecting pages. Each page has inlinks (incoming links) and outlinks (outgoing links). The number of inlinks of a page p is its indegree $\iota(p)$, and the number of outlinks is its outdegree $\omega(p)$.

4.2 Matrix Representations

Two matrix representations are introduced:

- Transition matrix T: Represents the probability of transitioning from one page to another. $T(p, q) = 0$ if there is no link from q to p, and $T(p, q) = \frac{1}{\omega(q)}$ if there is a link from q to p.
- Inverse transition matrix U: Represents the probability of transitioning from one page to another in the reverse direction. $U(p, q) = 0$ if there is no link from p to q, and $U(p, q) = \frac{1}{\iota(q)}$ if there is a link from p to q.

4.3 PageRank Algorithm

PageRank assigns global importance scores to pages based on link information. The PageRank score $r(p)$ of a page p is calculated using the formula:

$$r(p) = \alpha \cdot \sum_{q:(q,p)\in E} \frac{r(q)}{\omega(q)} + (1-\alpha) \cdot \frac{1}{N}$$

PageRank scores are computed iteratively using the Jacobi method with a fixed number of iterations or until convergence.

4.4 Trust Function and TrustRank Algorithm

The objective is to estimate the likelihood that a page is good without invoking an oracle function for all pages. A trust function T yields values between 0 (bad) and 1 (good). Ideally, $T(p)$ should give the probability that page p is good. An empirical observation is made about the approximate isolation of the good set: good pages seldom point to bad ones. The TrustRank algorithm is developed to estimate trust scores for pages, integrating PageRank with trust information. The TrustRank score $r(p)$ of a page p is computed iteratively using the formula:

$$r = \alpha \cdot T \cdot r + (1-\alpha) \cdot U \cdot O$$

where O is the oracle score vector indicating if a page is good or bad.

4.5 Ignorant Trust Function (Baseline)

To evaluate pages without invoking the oracle function for all pages, a baseline trust function T_0 is introduced. In the Ignorant Trust Function, a limited budget of oracle invocations is used to select a seed set of pages, and all other pages are assigned a trust score of $\frac{1}{2}$.

High Level explanation of our code:

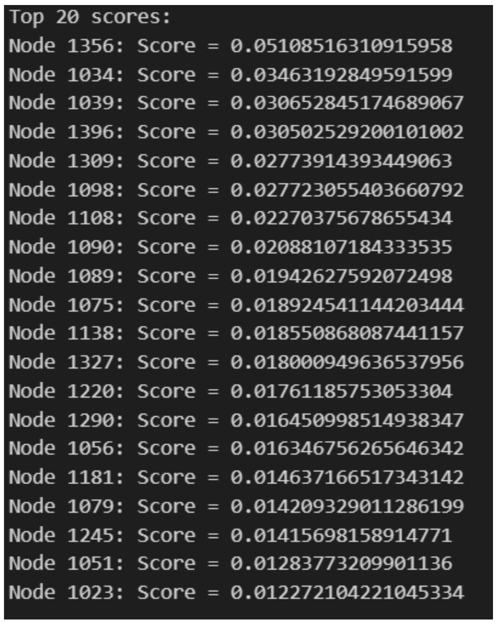
1. **Reading and preprocessing data:** Extract and preprocess payment data and information about potentially unreliable senders.
2. **Creating Transition Matrix (T):** Construct a transition matrix representing the probability of transitioning from one node to another based on payment data.
3. **Creating Inverse Transition Matrix (U):** Generate an inverse transition matrix to represent the probability of transitioning.
4. **PageRank Algorithm:** Compute PageRank scores using the transition matrix to assign global importance scores to each node.
5. **TrustRank Algorithm:** Implement the TrustRank algorithm, integrating PageRank with trust information to estimate trust scores for each node.
6. **visualization:** Visualize the results, including top and bottom scores, and store the scores in a results file(sorted order).

Algorithm 1: The TrustRank algorithm

```
input  : T transition matrix, $N$ number of pages, $L$ limit of oracle invocations,
         $\alpha_B$ decay factor for biased PageRank, $M_B$ number of biased
         PageRank iterations
output: $t^*$ TrustRank scores
// evaluate seed-desirability of pages
$s \leftarrow$ SelectSeed();
// generate corresponding ordering
$\sigma \leftarrow$ Rank$(\{1, \ldots, N\}, s)$;
// select good seeds
$d \leftarrow 0_N$;
for $i \leftarrow 1$ to $L$ do
    if $O(\sigma(i)) == 1$ then
        $d(\sigma(i)) \leftarrow 1$;
    end
end
// normalize static score distribution vector
$d \leftarrow d/\|d\|$;
// compute TrustRank scores
$t^* \leftarrow d$;
for $i \leftarrow 1$ to $M_B$ do
    $t^* \leftarrow \alpha_B \cdot T \cdot t^* + (1 - \alpha_B) \cdot d$;
end
return $t^*$;
```

5 Results

The Bad Scores is stored in the output file ’results.txt’, providing a trust(bad) scores of nodes.Figure 2 illustrates the top 20 nodes with the highest bad scores. Remarkably, among the nodes evaluated, approximately 146 nodes exhibit a pristine record, indicated by a bad score of 0. These nodes epitomize the ideal entities within the network, demonstrating no association with bad nodes. This finding underscores the presence of nodes that maintain a commendable distance from bad connections. Figure 3 presents the bad scores of nodes, while Figure 4 depicts the distribution of bad scores.



```
Top 20 scores:
Node 1356: Score = 0.05108516310915958
Node 1034: Score = 0.03463192849591599
Node 1039: Score = 0.030652845174689067
Node 1396: Score = 0.030502529200101002
Node 1309: Score = 0.02773914393449063
Node 1098: Score = 0.027723055403660792
Node 1108: Score = 0.02270375678655434
Node 1090: Score = 0.02088107184333535
Node 1089: Score = 0.01942627592072498
Node 1075: Score = 0.018924541144203444
Node 1138: Score = 0.018550868087441157
Node 1327: Score = 0.018000949636537956
Node 1220: Score = 0.01761185753053304
Node 1290: Score = 0.016450998514938347
Node 1056: Score = 0.016346756265646342
Node 1181: Score = 0.014637166517343142
Node 1079: Score = 0.014209329011286199
Node 1245: Score = 0.01415698158914771
Node 1051: Score = 0.01283773209901136
Node 1023: Score = 0.012272104221045334
```

Figure 2: Top 20 Nodes(Sender/Receiver) with their bad scores

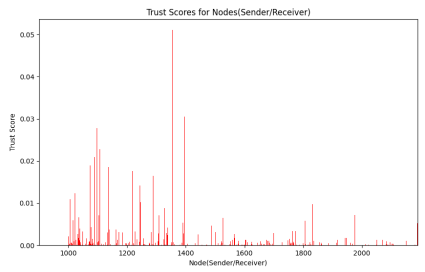


Figure 3: bad scores of nodes(Sender/Receiver)

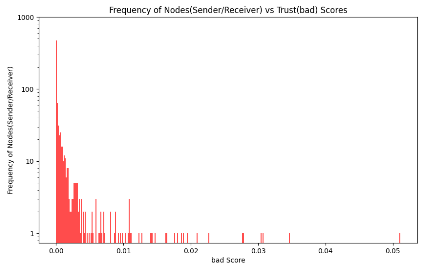


Figure 4: Frequency Distribution of scores