EigenTrust
The EigenTrust Algorithm for Reputation Management in P2P Networks

Sepandar D. Kamvar
Mario T. Schlosser
Hector Garcia-Molina

Stanford University
Problem

- **Problem:**
  - Reduce inauthentic files distributed by malicious peers on a P2P network.

- **Motivation:**
  “Major record labels have launched an aggressive new guerrilla assault on the underground music networks, flooding online swapping services with bogus copies of popular songs.”

  - Silicon Valley Weekly
Problem

- **Goal:** To identify sources of inauthentic files and bias peers against downloading from them.

- **Method:** Give each peer a *trust value* based on its previous behavior.
Some approaches

- Past History
- Friends of Friends
- EigenTrust
Terminology

- **Local trust value**: $c_{ij}$. The opinion that peer $i$ has of peer $j$, based on past experience.
- **Global trust value**: $t_i$. The trust that the entire system places in peer $i$. 

![Diagram showing peer trust values and interactions]

- $t_1 = 0.3$
- $t_2 = 0.2$
- $t_3 = 0.5$
- $t_4 = 0$
- $c_{12} = 0.3$
- $c_{21} = 0.6$
- $c_{23} = 0.7$
- $c_{14} = 0.01$
Local Trust Values

- Each time peer $i$ downloads an authentic file from peer $j$, $c_{ij}$ increases.
- Each time peer $i$ downloads an inauthentic file from peer $j$, $c_{ij}$ decreases.

\[ c_{ij} = \text{[Image of trust assessment]} \]
Normalizing Local Trust Values

- All $c_{ij}$ non-negative
- $c_{i1} + c_{i2} + \ldots + c_{in} = 1$
Local Trust Vector

- **Local trust vector** $c_i$: contains all local trust values $c_{ij}$ that peer $i$ has of other peers $j$.

\[
\begin{pmatrix}
0 \\
0 \\
0 \\
0.1
\end{pmatrix} = \begin{pmatrix}
0 \\
0.9 \\
0 \\
0.1
\end{pmatrix}
\]

Peer 1

Peer 2

Peer 3

Peer 4
Some approaches

- Past History
- Friends of Friends
- EigenTrust
Past history

- Each peer biases its choice of downloads using its own opinion vector $c_i$.
- If it has had good past experience with peer $j$, it will be more likely to download from that peer.
- Problem: Each peer has limited past experience. Knows few other peers.
Friends of Friends

- Ask for the opinions of the people who you trust.
Friends of Friends

- Weight their opinions by your trust in them.
The Math

\[ C'_{ik} = \sum_j C_{ij} \cdot C_{jk} \]

What they think of peer k.

Ask your friends j

And weight each friend's opinion by how much you trust him.

\[ \begin{array}{cccccc}
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
0.2 & 0.3 & 0.5 & 1 & 0 & 0 \\
\end{array} \]

\[ c'_i = C^T c_i \]
Problem with Friends

- Either you know a lot of friends, in which case, you have to compute and store many values.
- Or, you have few friends, in which case you won’t know many peers, even after asking your friends.
Dual Goal

- We want each peer to:
  - Know all peers.
  - Perform minimal computation (and storage).
Knowing All Peers

- Ask your friends: 
  \[ t = C^T c_i. \]
- Ask their friends: 
  \[ t = (C^T)^2 c_i. \]
- Keep asking until the cows come home: 
  \[ t = (C^T)^n c_i. \]
Minimal Computation

- Luckily, the trust vector $t$, if computed in this manner, converges to the same thing for every peer!
- Therefore, each peer doesn’t have to store and compute its own trust vector. The whole network can cooperate to store and compute $t$. 
Non-distributed Algorithm

- Initialize:
  \[ t^{(0)} = \left[ \frac{1}{n} \quad \ldots \quad \frac{1}{n} \right]^T \]

- Repeat until convergence:
  \[ t^{(k+1)} = C^T t^{(k)} \]
Distributed Algorithm

- No central authority to store and compute $t$.
- Each peer $i$ holds its own opinions $c_i$.
- For now, let’s ignore questions of lying, and let each peer store and compute its own trust value.

$$t_i^{(k+1)} = c_{i1}t_1^{(k)} + \ldots + c_{in}t_n^{(k)}$$
Distributed Algorithm

For each peer i {
    -First, ask peers who know you for their opinions of you.
    -Repeat until convergence {
        -Compute current trust value: $t_i^{(k+1)} = c_{1j} t_1^{(k)} + ... + c_{nj} t_n^{(k)}$
        -Send your opinion $c_{ij}$ and trust value $t_i^{(k)}$ to your acquaintances.
        -Wait for the peers who know you to send you their trust values and opinions.
    }
}
Probabilistic Interpretation
Malicious Collectives
Pre-trusted Peers

- Battling Malicious Collectives
- Inactive Peers
- Incorporating heuristic notions of trust
- Convergence Rate
Pre-trusted Peers

- Battling Malicious Collectives
- Inactive Peers
- Incorporating heuristic notions of trust
- Convergence Rate
Secure Score Management

- Two basic ideas:
  - Instead of having a peer compute and store its own score, have another peer compute and store its score.
  - Have multiple score managers who vote on a peer’s score.
How to use the trust values $t_i$

- When you get responses from multiple peers:
  - Deterministic: Choose the one with highest trust value.
  - Probabilistic: Choose a peer with probability proportional to its trust value.
Load Distribution

Deterministic Download Choice

Probabilistic Download Choice
Threat Scenarios

- **Malicious Individuals**
  - Always provide inauthentic files.

- **Malicious Collective**
  - Always provide inauthentic files.
  - Know each other. Give each other good opinions, and give other peers bad opinions.
More Threat Scenarios

- **Camouflaged Collective**
  - Provide authentic files some of the time to trick good peers into giving them good opinions.

- **Malicious Spies**
  - Some members of the collective give good files all the time, but give good opinions to malicious peers.
Malicious Individuals

![Bar chart showing the fraction of inauthentic downloads against the fraction of malicious peers. The chart compares non-trust based and trust based methods.]
Malicious Collective

![Chart showing the fraction of inauthentic downloads versus the fraction of malicious peers. The chart compares non-trust based and trust based methods.]
Camouflaged Collective
Malicious Spies

![Graph showing inauthentic downloads vs. authentic malicious uploads]

- Lines for trusted and non-trusted categories.
Conclusion

- Eigentrust
  - Dramatically reduces number of inauthentic files on the network.
  - Robust to malicious peers.
  - Low overhead.
The End