Reputation, Trust and Recommendation Systems in Peer-to-Peer Environments

Boaz Patt-Shamir
Tel Aviv University
Consider eBay.

- Successful e-commerce!
  - More than 40,000,000 items listed at any moment
  - 2006 sales: $52.5B
- Real concern: Why trust virtual sellers/buyers?!
- At least partly thanks to their reputation system.
  - System maintains a record for each user on a public “billboard”
  - After each transaction, each party ranks the other
  - Users with too many bad opinions are practically dead.
Suppose I Want A Blackberry
Let's Check This Jgonzo Out!

Member Profile: jgonzo166 (23 ⭐ )

Feedback Score: 23
Positive Feedback: 92.6%

- Members who left a positive: 25
- Members who left a negative: 2
- All positive feedback received: 25

Recent Ratings:

<table>
<thead>
<tr>
<th></th>
<th>Past Month</th>
<th>Past 6 Months</th>
<th>Past 12 Months</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>2</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>neutral</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>negative</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Bid Retractions (Past 6 months): 0

Member since: Nov-13-05
Location: United States
- ID History
- Items for Sale
- Add to Favorite Sellers
- View My World Page
- View my Reviews & Guides

Contact Member

Feedback Received

From Buyers From Sellers Left for Others

27 feedback received by jgonzo166 (0 ratings mutually withdrawn)

<table>
<thead>
<tr>
<th>Comment</th>
<th>From</th>
<th>Date / Time</th>
<th>Item #</th>
</tr>
</thead>
<tbody>
<tr>
<td>very smooth transaction, the part was payed for and picked up quickly</td>
<td>Seller stevenp6198 (17 ⭐ )</td>
<td>Mar-06-07 17:03</td>
<td>120091505427</td>
</tr>
<tr>
<td>great ebayer will do business with again</td>
<td>Seller pa1955 (902 ⭐ )</td>
<td>Feb-28-07 17:40</td>
<td>140088660855</td>
</tr>
</tbody>
</table>
How about some spam?

Once a message is known to be spam—filter kills it.

Main problem: Who’s to say what’s spam?

• Usually: questionable heuristics

• SPAMNET, gmail: humans mark spam email
  - Marks are distributed to client filters
  - Work amortized over user population

• Vulnerable to spammers!

• Solution: rank user’s trustworthiness
  - How? Well...
Simple Model

• \( n \) players
  - \( \alpha \cdot n \) of them are honest
  - the rest may be arbitrarily malicious (Byzantine)

• \( m \) objects
  - each object has known cost, unknown value
  - say \( \beta \cdot m \) of the objects are good

• Execution proceeds in rounds. Each player:
  - Reads billboard
  - Probes an object (incur its cost!)
  - Posts result
Talk Overview

✓ Introduction

• Simple approaches
• A Lower bound
• A simple algorithm
• The “who I am” problem
• Extensions
Let’s be concrete

Assume (for the time being) that...

• $n \approx m$ and both are large
• All objects have cost 1
• There is only one object of value 1, and all other objects have value 0
• The goal is that honest players find the good object.
**Attempt 1: Think Positive!**

Rule: Always try the object with the highest number of good recommendations.

- Adversarial strategy: recommend bad objects.
  - Honest players will try all bogus recommendations before giving them up!
  - $\Omega(n)$ cost per honest player.
**Attempt 2: Risk Averse**

Rule: Always probe the object with the least number of negative recommendations

- Adversarial strategy: slander the good object
  - Each player will try all other objects first!
  - Again, $\Omega(n)$ cost per honest player.

- (Very popular policy, but very vulnerable)
Attempt 3: A Combination?

Rule: Always probe the object with the largest “net recommendation” $\equiv$ positive - negative.

- Adversarial strategy: recommend some bad objects, slander the good object
  - Can still force $\Omega(n)$ cost per honest player.
Let’s get fancy: Trust!

- Idea: assign trust value to each player
- Direct assignment: based on agreements and conflicts.
- Take the “transitive closure”: how?
- Use algorithms for web searches to find “consensus” trust value for each player
  - PageRank [Google]: steady-state probability
  - HITS [Kleinberg]: left, right eigenvectors (hubs & authorities)
- Larger weight to opinionated players, and to players many opine about
Transitive Trust Fails

- Algorithms find “vox populi”, but is this what we’re after?
- Tightly-knit community: A clique of well-coordinated crooks can overtake the popular vote!
- Result: discourage honest players to voice their opinion, for fear of being discredited.
- Empirical study [WWW’2003] required a priori “trusted authorities”...
Find A Good Object: Some Results

- No algorithm can stop in less than $\Omega\left(\frac{1}{\alpha}\right)$ rounds
- Synchronous algorithm that stop in expected $O\left(\frac{1}{\alpha} \cdot \frac{\log n}{\log \log n}\right)$ rounds
- Asynchronous algorithm with total work $O(n \log n)$
- Extensions:
  - Unknown desired value
  - Competitive algorithm for dynamic objects
  - Competitive algorithm for users with different interests/availabilities
A Simple Lower Bound

Theorem: For any algorithm and player there exists a scenario where the expected number of probes the player makes is $\Omega \left( \frac{1}{\alpha} \right)$. 

Proof: By symmetry. Consider $\frac{1}{\alpha}$ groups of players, each running the alg., claiming a different object to be the best. Must go and check...

Ours is the good one!

No, ours is!
A SIMPLE ALGORITHM

• If I’m the only honest guy, I must try all objects.
  - Will take $\Omega(n)$ probes

• If there all others are honest, heed their advice
  - But what about crooks?

• Balanced rule: With probability $\frac{1}{2}$, try a random object; and with probability $\frac{1}{2}$, follow a random advice.

Theorem: If all honest players follow the balanced rule, they will all find the good object in $O\left(\frac{\log n}{\alpha}\right)$ expected rounds.
**Analysis of Simple Algorithm**

Consider the execution into three parts, according to the number of votes on the good object.

1. No votes for good object.
2. At most $\alpha n/2$ votes for good object.
3. More than $\alpha n/2$ votes for good object.

**Part 1:**
- $\textbf{Prob}[\text{random object is good}] = 1/n$
- In each round: appx. $\alpha n/2$ random objects probed
- Good object found in expected $O(1/\alpha)$ rounds

$O(1/\alpha)$ rounds
Analysis of Simple Algorithm

Consider the execution into three parts, according to the number of votes on the good object.

1. No votes for good object. \( O(1/\alpha) \) rounds

2. At most \( \alpha n/2 \) votes for good object. \( O(\log n/\alpha) \) rounds

3. More than \( \alpha n/2 \) votes for good object.

Part 2: assume there are \( k > 0 \) votes for good object. Then

- \( \text{Prob}[\text{random advice is good}] = k/n \)
- In a round: appx. \( \alpha n/4 \) random advices followed
  - Expected \# good votes after round: \( k + k\alpha/4 \)
  - \( O\left(\frac{\log n}{\alpha}\right) \) rounds until majority is satisfied
Analysis of Simple Algorithm

Consider the execution into three parts, according to the number of votes on the good object.

1. No votes for good object. \( O(1/\alpha) \) rounds
2. At most \( \alpha n/2 \) votes for good object. \( O(\log n/\alpha) \) rounds
3. More than \( \alpha n/2 \) votes for good object. \( O(1/\alpha) \) rounds

Part 3: Consider a single player.

- \textbf{Prob} [ random advice is good ] \( \geq \alpha/2 \)
  - Expected \# random advices until player hits a good one: \( O(1/\alpha) \)
**Analysis of Simple Algorithm**

Consider the execution into three parts, according to the number of votes on the good object.

1. No votes for good object. \( O(1/\alpha) \) rounds
2. At most \( \alpha n/2 \) votes for good object. \( O(\log n/\alpha) \) rounds
3. More than \( \alpha n/2 \) votes for good object. \( O(1/\alpha) \) rounds

**Total expected rounds:** \( O(\log n/\alpha) \) rounds

Works also asynchronously: \( O(n \log n) \) total work for the honest guys
Implication: P2P Web Search

- Currently, web search is centralized
  - client-server model: vulnerable!
- Suppose some peers are looking for something
  - algorithm: try a page or try a recommendation
  - even if only $\alpha$ fraction are honestly following protocol, they will all find the result in $O(\log n/\alpha)$ rounds
What if not all honest users agree?

- Post-modern world: every view is legitimate
- Every taste group should collaborate
- Who is in my taste group?
- New goal: Reveal complete preference vector
- Suppose that each player knows $0 < \alpha \leq 1$ such that at least an $\alpha$ fraction of the players share his exact same taste.
THE “WHO I AM” PROBLEM

• Motivation:
  - If I’m looking for T objects, must I pay T/α?
  - No, I must pay only 1/α + T.
  - Need to identify who can I rely on

• Abstraction:
  - Each user has his preference vector
  - Users with identical vectors belong to the same taste group
  - Goal: find the complete preference vector
Algorithm Finding Who I Am

Main idea: Given a set of players and a set of objects:
- Randomly split the players and objects into two subsets and assign each player subset to an object subset
- Each player subset recursively solves its object subset
- Then results are merged
**Example: We start with a matrix of players and objects**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Boaz Patt-Shamir
LADIS 2007
**Split the players and objects into 2 sections**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>o1</td>
<td>o2</td>
<td>o3</td>
<td>o4</td>
<td>o5</td>
<td>o6</td>
<td>o7</td>
<td>o8</td>
<td>o9</td>
<td>a10</td>
<td>a11</td>
<td>a12</td>
<td>a13</td>
<td>a14</td>
<td>a15</td>
<td>a16</td>
</tr>
</tbody>
</table>
**If size of set not small enough - split again**
If size of set small enough - probe all objects
Now estimate the rest of your vector and return.
**Estimate the rest of your vector and return**
How to merge?

- My taste has popularity $\alpha$ (whp in each subset)
- Sufficient to choose from vectors of popularity at least $\alpha/2$ (say)
- Probe objects of disagreement: at least one vector eliminated
- $O(1/\alpha)$ probes per recursion level
  $\Rightarrow O(\log n/\alpha)$ probes overall
Conclusion & Open Problems

- Good news: Can find who I am close to best possible at given budget
- ... but far from being completely solved:
  - Can $1/\alpha$ factor in approximation be removed?
  - Is there an asynchronous algorithm?
  - Can communication cost be reduced?
  - How about non-binary grades?

“tell me who are your friends, and I’ll tell you who you are”
Thank you!

...and to my co-authors:
Alon, Awerbuch, Azar, Lotker, Nisgav, Peleg, Tuttle