Securing Social Media: A Network Structure Approach

Nitin Chiluka
Securing Social Media: A Network Structure Approach

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 4 juni 2013 om 12.30 uur

doors Nitin Janardhan CHILUKA

Master of Technology in Information Technology, Atal Bihari Vajpayee - Indian Institute of Information Technology and Management, Gwalior, India
geboren te Hyderabad, India
Dit proefschrift is goedgekeurd door de promotor:
Prof.dr.ir. H.J. Sips

Samenstelling promotiecommissie:

Rector Magnificus...

Prof.dr.ir. H.J. Sips...
Dr.ir. J.A. Pouwelse...
Prof.dr. A. Mei...
Prof.dr.ir. H. Bos...
Prof.dr. C. Witteveen...
Dr.ir. J. van den Berg...
Dr. K.P. Gummadi...
Prof.dr.ir. D.H.J. Epema...

voorzitter
Technische Universiteit Delft, promotor
Technische Universiteit Delft, copromotor
University of Rome La Sapienza
Vrije Universiteit Amsterdam
Technische Universiteit Delft
Technische Universiteit Delft
Max Planck Institute
Technische Universiteit Delft, reservelid

Published and distributed by: Nitin Chiluka
E-mail: nitin.chiluka@gmail.com

Keywords: social network, sybil, defense, trust, distrust, reputation, recommendation.

Copyright © 2013 by Nitin Chiluka.

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission of the author.

Printed in The Netherlands by: Wöhrmann Print Service.

The work described in this thesis has been carried out in the ASCI graduate school. ASCI dissertation series number 282.

This work was supported by the European Community’s Seventh Framework Programme through the P2P-NEXT project (grant no. 216217) and the QLectives project (grant no. 231200).
Acknowledgments

My PhD would not have been possible without many people’s contributions along the way. Let me take the opportunity to thank a few who have influenced me in their own fashion.

Henk, I owe you a debt of gratitude for giving me the opportunity to do a PhD at TU Delft. You were very generous to turn my request for an internship in September 2007 into a PhD candidacy a year later. Your guidance particularly in the final year was critical and invaluable. While my research work may have been a bit different from your expertise, your comments during our meetings were very insightful.

Johan, thank you very much for the freedom you have given me during the last 4 years. Our diverse group of 15-20 researchers you brought together with two large EU projects is something you should be very proud of. I hope that you are instrumental in creating such groups again in the coming years.

Naza, you get my special shoutout. Thank you very much for guiding me throughout my PhD, in-person and on-skype. It was only through your invitation to Campina Grande that I had some of the best times in the last 5 years. Writing a paper in the day and attending the carnival in the night was a great combo. Wish Aida, you, and your hospitable family all the very best!

Dimitra and Boxun, thank you for inviting me to be your officemate after Naza left. I enjoyed talking to you both about every topic in the world (and alternate universes); and that was everyday. Dimitra, thank you very much for your graph expertise inputs on my ideas. Boxun, I enjoyed our weekly dailyyok dinners at 7:45 pm. Wish you both the very best!

Mihai, visiting Brazil with you was fun. Rio de Janeiro was possible only with your broken Portuguese. On a different note, given your wit, you should consider writing for The Daily Show or SNL :-)

Rahim, I learned tennis much better playing with you than from a coach. I also enjoyed going on a day trip to places like Texel and Brugge. I liked our many dinners at Burgerz with Boxun.

Lucia, I enjoyed talking to you be it in the office corridor, Aula or gtalk. I always admired your straightforwardness on any issue. I also liked that you took over from Victor to bring muffins for post-lunch.

Tamas, you are one of the most humble and talented persons I know. I would like to
thank you for collaborating with me on the parallel computing lab course for 3 years. Wish you a great time in Hungary with your family.

Rameez, thank you for setting the bar very high for our group. Besides research, I enjoyed our conversations about cricket and world affairs in Hinglish. Yes Tamas, its a real language ;-) 

Riccardo, its always fun talking to you over espressos at Aula.
Adele, if your PhD is any indication, you’ll have a wonderful academic career.
Niels, thanks for CPR-ing channels in Tribler with great GUI work.
Arno and Boudewijn, Tribler could not have sustained for so long without your persistence. Thanks a lot on behalf of the group.

Paulo, Munire, and Stephen, thanks so much for DAS and ICT support as well as helping Tamas and me during in4026 lab course by assigning computers and accounts.
Ilse and Rina, thank you for helping me on HR-related issues.

Praveen, you are one special brother. Your journey has always been a great source of inspiration to me. Wish you, Sneha (my sister-in-law), and Meenu (my niece) all the very best!

Mom and Dad, I am ever so grateful to you for being always there for me, rooting for me every step of the way. You brought me up with a strong sense of independence, which rubs on almost every aspect of my life. That I believe is probably the best education you gave me. Thank you very much for everything.
# Contents

## Acknowledgments i

1 Introduction 1

1.1 Online Social Media ................................................. 2
   1.1.1 Evolution .................................................. 2
   1.1.2 Magnitude .................................................. 3

1.2 State of the Union .................................................... 3
   1.2.1 Network Structure Analysis .................................. 4
   1.2.2 Sparsity and Scalability ..................................... 4
   1.2.3 Adversarial Environment ..................................... 6
   1.2.4 The Importance of Distrust ................................... 7
   1.2.5 Decentralization ............................................. 9

1.3 Research Questions ................................................... 9

1.4 Contributions and Thesis Outline .................................. 11

2 A Link Prediction Approach to Recommendations in Large OSM 13

2.1 Background and Related Work .................................... 15
   2.1.1 Online Social Media ....................................... 15
   2.1.2 Item Recommendations in OSM .............................. 15
   2.1.3 Social Influence in OSM ..................................... 16

2.2 A Link Prediction Approach to Collaborative Filtering .......... 16
   2.2.1 Notation .................................................... 17
   2.2.2 Algorithms ................................................ 17

2.3 Experimental Methodology .......................................... 19
   2.3.1 Dataset .................................................... 19
   2.3.2 Data Collection ............................................ 19
   2.3.3 Dataset Description ........................................ 19
   2.3.4 Metrics ................................................... 20
   2.3.5 Baseline Method ........................................... 20

2.4 Evaluation .......................................................... 20
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4.1 Link Prediction Performance</td>
<td>21</td>
</tr>
<tr>
<td>2.4.2 Influence of Explicit Social Links</td>
<td>23</td>
</tr>
<tr>
<td>2.4.3 Influence of User Behavior</td>
<td>24</td>
</tr>
<tr>
<td>2.5 Conclusions</td>
<td>25</td>
</tr>
<tr>
<td>3 Personalizing EigenTrust in the face of Centrality Attack</td>
<td>27</td>
</tr>
<tr>
<td>3.1 Analysis of EigenTrust</td>
<td>28</td>
</tr>
<tr>
<td>3.1.1 Algorithmic overview of ET</td>
<td>29</td>
</tr>
<tr>
<td>3.1.2 ET as a community detection algorithm</td>
<td>29</td>
</tr>
<tr>
<td>3.1.3 Eigenvector centrality attack &amp; its impact</td>
<td>30</td>
</tr>
<tr>
<td>3.1.4 Limitations</td>
<td>32</td>
</tr>
<tr>
<td>3.2 Personalized EigenTrust</td>
<td>32</td>
</tr>
<tr>
<td>3.2.1 Approach</td>
<td>32</td>
</tr>
<tr>
<td>3.2.2 Basic computation</td>
<td>33</td>
</tr>
<tr>
<td>3.2.3 Scalable computation</td>
<td>34</td>
</tr>
<tr>
<td>3.3 Simulation Setup</td>
<td>35</td>
</tr>
<tr>
<td>3.4 Resilience to attacks</td>
<td>36</td>
</tr>
<tr>
<td>3.4.1 Collusion with malicious spies</td>
<td>36</td>
</tr>
<tr>
<td>3.4.2 Eigenvector centrality attack</td>
<td>36</td>
</tr>
<tr>
<td>3.4.3 Extended Traitor attack</td>
<td>38</td>
</tr>
<tr>
<td>3.4.4 Impact on transaction models</td>
<td>38</td>
</tr>
<tr>
<td>3.4.5 Sensitivity to teleportation parameter ( \alpha )</td>
<td>40</td>
</tr>
<tr>
<td>3.4.6 Impact on ranking of search results</td>
<td>40</td>
</tr>
<tr>
<td>3.5 Role of Social Network</td>
<td>41</td>
</tr>
<tr>
<td>3.5.1 Bootstrapping into the network</td>
<td>41</td>
</tr>
<tr>
<td>3.5.2 Defending against Sybil attacks</td>
<td>42</td>
</tr>
<tr>
<td>3.6 Related Work</td>
<td>43</td>
</tr>
<tr>
<td>3.7 Concluding Remarks</td>
<td>44</td>
</tr>
<tr>
<td>4 Leveraging Trust and Distrust for Sybil-Tolerant Voting in Online Social Media</td>
<td>47</td>
</tr>
<tr>
<td>4.1 Background and Related Work</td>
<td>48</td>
</tr>
<tr>
<td>4.1.1 Social Network-based Sybil Defenses</td>
<td>48</td>
</tr>
<tr>
<td>4.1.2 Signed Network Analysis</td>
<td>50</td>
</tr>
<tr>
<td>4.2 Preliminaries</td>
<td>51</td>
</tr>
<tr>
<td>4.2.1 Notations and Definitions</td>
<td>51</td>
</tr>
<tr>
<td>4.2.2 Datasets and Mixing Times</td>
<td>51</td>
</tr>
<tr>
<td>4.3 Method</td>
<td>51</td>
</tr>
<tr>
<td>4.3.1 System Model and Approach</td>
<td>53</td>
</tr>
<tr>
<td>4.3.2 Resistance Network</td>
<td>55</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>6.2</td>
<td>Effect of Churn on a Trust Overlay</td>
</tr>
<tr>
<td>6.3</td>
<td>Improving Connectivity</td>
</tr>
<tr>
<td>6.3.1</td>
<td>Two-hop Neighborhood</td>
</tr>
<tr>
<td>6.3.2</td>
<td>Sybil Attack</td>
</tr>
<tr>
<td>6.4</td>
<td>Adaptive 2-Hop Design</td>
</tr>
<tr>
<td>6.4.1</td>
<td>Goal and Assumptions</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Approach Overview</td>
</tr>
<tr>
<td>6.4.3</td>
<td>Method</td>
</tr>
<tr>
<td>6.5</td>
<td>Trace-Driven Evaluation</td>
</tr>
<tr>
<td>6.5.1</td>
<td>Experimental Setup</td>
</tr>
<tr>
<td>6.5.2</td>
<td>Tradeoff: Robustness vs Connectivity</td>
</tr>
<tr>
<td>6.5.3</td>
<td>Steady State</td>
</tr>
<tr>
<td>6.5.4</td>
<td>Flashcrowd</td>
</tr>
<tr>
<td>6.5.5</td>
<td>Recession</td>
</tr>
<tr>
<td>6.6</td>
<td>Related Work</td>
</tr>
<tr>
<td>6.6.1</td>
<td>Decentralized Social Networks</td>
</tr>
<tr>
<td>6.6.2</td>
<td>Social Networks &amp; Distributed Systems</td>
</tr>
<tr>
<td>6.6.3</td>
<td>P2P Churn</td>
</tr>
<tr>
<td>6.7</td>
<td>Conclusion</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
</tr>
<tr>
<td>7.1</td>
<td>Conclusions</td>
</tr>
<tr>
<td>7.2</td>
<td>Future Work</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
</tr>
<tr>
<td></td>
<td>Samenvatting</td>
</tr>
<tr>
<td></td>
<td>Curriculum vitae</td>
</tr>
</tbody>
</table>
Online social media (OSM) is a natural offspring of the democratization of the Internet. Any user can publish and share messages, blogs, photos, videos, and other media with others. Users can form friendships with others as well as follow them to keep updated. Such freedom coupled with low cost of content production, consumption, and distribution has enabled the growth of OSM to today’s unprecedented scales. For instance, current popular OSM such as Facebook, Twitter, and YouTube have more than 100 million active users every month publishing millions of photos, messages and videos on their websites daily [3, 10, 15].

A vast user base and a wealth of content in OSM, however, presents its own challenges, some of which we outline here. First, the amount of user-generated content being uploaded in their repositories [30] makes it non-trivial for users to find relevant content [19]. Second, the ease of creating an account and participating in OSM enables malicious users to spread and promote spam content while degrading the experience for normal users [29, 46, 102]. Third, growing privacy concerns in OSM motivate users to move toward decentralized social networks [2, 26, 39] which are still in a nascent stage.

In this thesis, we propose a network structure approach [88, 98] to address these challenges in OSM. We build scalable and effective graph-based algorithms that recommend content personalized to each user. To tackle adversarial environments, we leverage the social network graph as well as negative feedback from normal users to limit the capability of knowledgeable attackers in multiple scenarios. We strongly argue the role of distrust in the form of negative feedback to be prominent for designing robust systems. Moreover, we highlight some of the limitations in existing efforts to build decentralized social networks. We explore a new design point in this field which may motivate practical designs for future social network-based distributed systems.
1.1 Online Social Media

Online Social Media (OSM) refer to an ecosystem of websites and tools that allow ordinary users to contribute and share their content items such as messages, blogs, bookmarked stories, photos, and videos with other users. OSM also enables users to form friendships with others as well as follow them to keep updated. In contrast, traditional media such as Disney, CNN, and New York Times employ specialists in making movies and reporting news stories on TV or in print. Although the content in traditional media may be more reliable and high quality, OSM offers a venue for normal users to express and disseminate their content and opinions to friends and wider audience at little cost. Since there are few regulations and little oversight in OSM, users experience a greater sense of freedom motivating them to produce, share, and consume more content.

1.1.1 Evolution

The current state of online social media (OSM) can be traced through two separate routes. The first route is online social networks (OSNs). While instant messaging services from AOL, Yahoo, and MSN in late 1990s enabled users to chat directly with their friends, the knowledge of users beyond their immediate friend circle was little. SixDegrees in 1997 was among the first social networks to tap into the potential of expanding this circle to friends of friends and beyond. Friendster is a social gaming site which allows users to maintain contacts of other users and share content with them. Friendster reached over 3 million users within months after launching in 2002. Following in the footsteps, Myspace (2003) and Facebook (2004) went onto attain more than 100 million users within a few years. While Myspace declined in recent years, Facebook continues its upward trend with a billion active users as of September 2012.

The second route to today’s OSM is user-generated content systems (UGCs). In LiveJournal which began in 1999, a user has a journal page where she may write blogs on which other users can comment. LiveJournal users were also encouraged to form friendships with other bloggers to keep updated of new blogs of each other. Slashdot (1997) and Epinions (1999) offer users to review and comment on technology stories and consumer products, respectively. Digg (2004) is a social news website where users publish links to stories from typically traditional media such as CNN and New York Times. Digg users can then vote on a story indicating its popularity. Flickr is a photo-sharing website started in 2004 where users can upload photos which others can view, bookmark and comment on. YouTube is a video-sharing website started in 2005 where users can upload videos which others can view, bookmark and vote like/dislike on them. Digg, Flickr and YouTube also allow users to follow others to keep abreast of new posts.

Nowadays, OSM is ubiquitous. Traditional news outlets and various websites allow
OSM buttons such as Facebook’s like, Twitter’s tweet, and Google’s +1 on their web pages using which users can share and bookmark the stories. In addition, user accounts of Facebook can be used to log into various websites which previously required their own registered accounts. Moreover, the once-independent social media outlets are increasingly becoming coherent with each other. A YouTube video, for instance, can be embedded into a Facebook user’s wall or any web page. A Facebook user can post on her wall as well as tweet the same message on Twitter simultaneously. Such a synergy allows users to communicate with others through various means.

1.1.2 Magnitude

The vast user base of OSM can be attributed to two main factors. First, the cost of joining the system is very low. Any user can create an account within a few minutes, and begin participating in the system immediately. Flickr, for instance, observed 51 million registered members in June 2011 [7]. As of February 2013 in YouTube, over 100 million users perform a social action such as comment on, like, and share videos [15]. In September 2012, over a billion users were seen active on Facebook with half of them using a mobile device [3]. Twitter has registered over 500 million users [10]. Many other OSM such as Digg, Slashdot and Myspace have tens of millions of users.

Second, the democratic nature of publishing and sharing content enables any user to upload and view large number of content items at little cost. Over 6 billion photos are being hosted by Flickr in August 2011 [5]. YouTube in February 2013 noted 72 hours of video being uploaded every minute, with 3 hours of video coming from mobile devices [15]. Facebook users upload over 250 million photos every day [3]. Twitter users send over 340 million tweets and issue over 1.6 million queries each day [10].

1.2 State of the Union

The unprecedented magnitude of users and content in OSM presents new challenges as well as new opportunities to actors at various levels: individual, community, corporate, government, and the human society as a whole. In this thesis, we focus at the individual level with an aim to improve user experience in adverse conditions of OSM. In particular, we examine the following challenges faced in OSM:

- **Sparsity & Scalability**: Is it feasible to find relevant content in vast repositories of OSM?

- **Adversarial Environment**: As a large user base attracts opportunistic and malicious users, how to build robust defense schemes that reduce the attack capability of an adversary while serving the honest users well?
• **Decentralization**: With growing research efforts in building privacy-preserving distributed social networks, how reasonable are their design considerations?

This section discusses these challenges in detail, and how the current state-of-the-art approaches tackle them.

### 1.2.1 Network Structure Analysis

Network structure analysis [88,98] helps in modeling and analyzing the complexity of user-user and user-item relationships in online social media. Consider a graph where a node represents either a user or an item, and an edge between two nodes represents a relationship between two users (friends) or a user and an item (preferences). One can study the properties of such a graph in a wide spectrum ranging from a local level to a global level. At a local level, a user node’s immediate neighborhood constitutes her friends as user nodes and her preferences as item nodes. At a global level, one can determine how important a particular node, be it a user or an item, is based on node’s position in the whole graph topology.

Given a node $p$, an intermediate level in the spectrum comprises a subgraph $S_p$ personalized around the node $p$ extending beyond its immediate neighborhood. Essentially, users in $S_p$ are friends of friends and acquaintances of $p$, and items in $S_p$ are probably of interest and relevance to $p$. The intuition is that (i) the closer a user node $q$ is to $p$, the more trustworthy $q$ is to $p$, and (ii) the closer an item node $i$ is to $p$, the more interesting and relevant $i$ is to $p$. Figure 1.1 depicts this intuition with an example of a small social network. Based on this fundamental rationale, trust schemes such as reputation systems and Sybil defenses as well as recommender systems leverage network structure for their effectiveness. In the rest of the chapter, we outline various challenges and review state-of-the-art approaches that employ this rationale.

### 1.2.2 Sparsity and Scalability

The democratic nature of publishing and sharing through OSM has attracted millions of users to contribute to these systems. Such large user bases, combined with the low cost and effort required to produce and publish content, typically create massive and fast-growing content repositories in OSM [30]. On the one hand, such sheer volume of content may cater to varied tastes of users. On the other hand, this scale gives rise to the problem of information overload, also formulated as the Babel objection [21]: differentiating quality from noise in such large user-generated data is very difficult. In essence, finding interesting and relevant content in OSM is a significant challenge.

Recommendation algorithms [42,58,68,87] are a candidate solution to address this problem. OSM can use these algorithms to suggest interesting and relevant content to users based on their past preferences, thereby partially automating the process of content discovery. In
Figure 1.1: [Best viewed in color.] Graph structure of a social network of 11 nodes. Given a node $p$, its immediate neighborhood consisting of $a$, $b$, and $c$ are $p$’s friends. Nodes $d$ and $e$ represent $p$’s acquaintances or friends of friends, which are considered less trustworthy compared to $p$’s friends $a - c$. Nodes $f - j$ which are further away from $p$ are among the least trustworthy from $p$’s perspective.

spite of this potential, however, few studies have explored recommending items in the context of OSM (with the exception of YouTube [19]). Collaborative filtering (CF) techniques have been established as a good fit for this purpose in editorially-generated content systems (Non-OSM), such as MovieLens and Netflix, but there is little or no evidence that these techniques would perform well if applied to OSM.

There are primarily two reasons why classical CF techniques which were found to be effective for Non-OSM – such as $k$-Nearest Neighbor – cannot be assumed to perform as well in OSM. First, the user-item matrix is substantially larger and sparser in OSM than in Non-OSM. This difference exacerbates the already existing limitations in CF techniques regarding scalability and sparsity [68]. While Non-OSM such as Internet Movie Database have about a million items [30], OSM may contain hundreds of millions of items. For instance, YouTube has recently claimed [14] that hundreds of thousands of videos are uploaded everyday on its website, accounting for 24 hours of videos every minute. Also, Flickr has celebrated the five billionth photo [1] uploaded in September 2010, and 3000 photos are being uploaded every minute.

The second reason that challenges the application of classical CF techniques in OSM relates to the lack of editorial control in these systems. While all items in Non-OSM are introduced and organized in respective categories by a few trusted editors, OSM contain content published by millions of users, and not all items are properly categorized or of good quality. This essentially magnifies the problem of users to sift through noise and find content
of interest and relevance.

Few studies have explored item recommendations in OSM. To the best of our knowledge, the work on YouTube by Baluja et al. [19] is the only extensive study on recommending items in OSM. The authors propose a novel graph-based technique called Adsorption, which recommends videos given a co-view graph representing which user viewed what video.

1.2.3 Adversarial Environment

The ease of creating an account in an online social network (OSN) such as Facebook, Flickr, and YouTube has mixed results. On the upside, little effort to join the network has contributed to its massive growth. Any user can register within a matter of a few minutes, and form friendships, follow others, and upload content such as photos and videos at little cost. On the downside, an adversary may perform a Sybil attack [43], by creating multiple fake accounts cheaply with a malicious intent to disrupt the system.

Real-world OSNs have observed Sybil attacks in their networks. According to Facebook’s filings [16] in August 2012, 83 million illegitimate accounts were found in the social network out of its 955 million active accounts. Since Facebook’s major revenue comes from targeted advertising, such a large fraction of fake accounts can result in loss or reduced spending of advertisers, thereby harming their business. Regional OSNs such as Renren [102] and Tuenti [29] also experience such fake accounts. Furthermore, link farming was found in Twitter [46] where fake accounts are used to follow each other.

Such attacks necessitate OSNs to adopt a wide range of defense schemes. Challenge-response mechanisms such as Captchas typically limit the rate at which Sybil accounts [9, 29, 92, 102] are created [25, 55]. However, cheap crowd-sourcing techniques enable fast account creation. Machine-learning techniques such as SVM can classify whether an account is malicious or not based on its properties such as friendship request and accept frequencies and clustering coefficient [102]. These techniques are effective in identifying Sybil accounts with behavior highly deviant from normal users. Nevertheless, a counter strategy by these accounts is to replicate the patterns of normal users which make it difficult for these techniques to differentiate honest users from malicious ones.

In addition to rate limiting techniques, one line of defense is a reputation system. Based on interactions and feedbacks from users, a reputation score for each user is computed which depicts the user’s trustworthiness. This enables an honest user to prejudge whether to perform an interaction with an unknown user based on the latter’s reputation. A good reputation system reduces the number and the cost of bad interactions, while ostracizing malicious users. Such a reputation system improves user experience. While reputation systems such as EigenTrust [57], PeerTrust [101], and PowerTrust [107] have been shown to perform well in P2P systems, their adoption in social networks has not been studied in detail. A recent study [29] in the field of social network-based Sybil defenses (SNSD) has shown
that EigenTrust performs better than many state-of-the-art SNSD schemes, indicating that P2P reputation systems can also be customized to address challenges in adversarial social media. This is primarily due to fundamental similarities in production and consumption in social media and P2P systems. For instance, the following two scenarios are similar: (i) a user uploading a video to YouTube which is streamed by a viewer, and (ii) a user uploading a file to a downloader in a P2P file-sharing network such as Tribler [83]. The recipient in both scenarios may rate whether the consumption was satisfactory or spam. This feedback enables fighting spam in the system.

Another line of defense is against Sybil attacks [43]. A large body of work leverages social networks by incorporating their properties such as inherent trust relationships among users and graph structure into the designs of social network-based Sybil defenses (SNSD) schemes [40, 65, 84, 91, 92, 104, 105]. Each of these schemes makes two fundamental assumptions. First, although an attacker can create an arbitrary number of identities, she cannot establish arbitrary number of trust relationships (attack edges) with honest users since forming a trust relationship requires high social engineering cost. This leads to a sparse cut between the Sybil region containing malicious identities and the non-Sybil/honest region containing honest users in the graph, which is then exploited by these schemes. Second, a social network graph is expander-like and fast-mixing [76] in that a random walk in the graph quickly reaches a stationary distribution. Hence, a short random walk starting from a node in the non-Sybil region rarely escapes into the Sybil region.

These schemes can be broadly classified into two categories: Sybil detection and Sybil tolerance, as characterized in [95]. Sybil detection schemes [40, 84, 91, 104, 105] solely depend on the graph structure to label nodes as Sybil or not. Although application-independent, these schemes run the risk of (i) false positives: an honest user is misclassified as a Sybil, thereby not granted any service, and (ii) false negatives: a Sybil is misclassified as a non-Sybil which may allow unwarranted and unlimited access. Sybil tolerance schemes [74, 82, 92], on the other hand, do not label nodes as Sybil or not. Instead, they limit the leverage of an attacker who may use multiple Sybil identities by exploiting both the graph structure and application-specific information.

1.2.4 The Importance of Distrust

The role of distrust is vital to critical systems. The more critical a system, the more important the role of distrust. A good critical system such as e-mail must offer users a guarantee that their mail is reliable and secure with no spam. Current e-mail services improve their spam detection techniques using negative feedback from users with features such as ‘report spam’. In contrast, less critical systems such as P2P file-sharing networks which depend mainly on user contribution offer users only positive incentives such as ‘tit-for-tat’. Disincentives typically increase the cost of participation, and hence systems such as BitTorrent
and Gnutella do not incorporate negative feedback.

OSNs have become highly critical in today’s world. Many users communicate important announcements by posting messages on their Facebook walls and tweeting on Twitter as events happen in real-time. The critical nature of such OSNs attracts spammers and malicious entities. OSNs such as Facebook enable ordinary users to report spam or abuse on many aspects: another user, an event, a group, a page, a photo, a video, and a message. While most spam and Sybil defense schemes proposed by the research community leverage only the trust relationships among users [49, 57, 65, 91, 104, 105], we believe distrust in the form of such negative feedback can also be incorporated to improve the robustness of these critical OSNs.

Trust and distrust in such systems are typically modeled as signed social networks (in short, signed networks). A trust relationship indicating friendship between two users is represented by a positive edge, while a distrust relationship suggesting antagonistic feelings toward one another is represented by a negative edge. Most OSNs keep the distrust relationships such as ‘report as spam/abuse’, unlike friendships, private. However, a few such as Slashdot make these negative links in the form of ‘foes’ and ‘freaks’ publicly accessible to the outside world.

Recent studies on signed networks have predominantly focused on the edge sign link prediction problem: given a social network with signs on all edges, how accurately can we determine the sign of a hidden edge? In one study, Leskovec et al. [63] employ social psychology theories of balance and status to predict the signs of edges. In another study [62], they develop a machine learning approach based on various combinations of triads for sign prediction. Kunegis et al. propose a number of spectral analysis techniques for sign prediction [60], and later extend them to clustering and visualization [61] using Laplacian matrices of the signed graphs. More recently, DuBois et al. [44] propose a promising technique for sign prediction which combines path-probability trust inference algorithm and spring embedding to infer network distance between nodes. Their rationale is that a node-pair with a positive edge attract each other, while a negative edge makes them repel.

Propagation of distrust, unlike trust, is a tricky issue. For instance, an enemy of an enemy is not necessarily a friend, while a friend of a friend can be considered trustworthy. Guha et al. [48] propose that trust can propagate multiple steps whereas distrust only propagates a single step. Kerchove et al. [41] propose PageTrust, along the lines of PageRank, incorporating the knowledge of negative links while performing the random walk. In addition, various other studies explored propagating distrust in the network [24, 72, 108]. However, none of the studies in the field of signed networks discussed their application in the context of attacks. We believe the role of distrust in defense to be prominent.
1.2.5 Decentralization

The centralized nature of OSNs enables tracking and censorship of users with relative ease. Curious or malicious entities can study a particular community of users, infer personalities from their individual usage behavior, track their locations even when offline, and censor their content. Such repeated leaks only exacerbates growing concerns of privacy among users.

Privacy of users can be compromised even with a noble intent. Researchers often crawl OSNs to collect datasets for their studies. Although researchers typically anonymize datasets when they share, attackers can still infer a lot about the users in the shared datasets with additional information. For instance, if an adversary forms a distinct topology in the original network, the adversary can identify individuals around its neighborhood in an anonymized graph \[18, 106\]. A recent study \[79\] showed that a third of users having accounts in both Twitter and Flickr can be identified given an anonymized Twitter graph with only 12% error rate, although common members constitute less than 15%. Many such re-identification cases have been studied in the fields of graphs, recommendations, and databases. Though there are many recent studies that propose stronger anonymization techniques, privacy concerns of OSN users remain.

Decentralized social networks offer an alternative to OSNs, where users hold all the ownership of their data as well as perform secure communication with their friends without passing any information through a central entity. Diaspora \[2\] is a prime example where users can setup their own servers to host content, form friends, share updates and multimedia content with others. Safebook \[39\] adopts a peer-to-peer architecture and real-world trust relationships to build a privacy-preserving social network. Peerson \[26\] is another effort that builds on a peer-to-peer infrastructure. Peerson users keep control of their data through encryption, key management and access control in a decentralized setting.

Another trend is that the trust inherent among users in online social networks is leveraged to build robust and secure distributed systems. A plausible argument – creating new trust relationships is expensive due to high social engineering costs – inspired the designs of generic decentralized Sybil defenses such as SybilGuard \[105\], SybilLimit \[104\], and Gatekeeper \[91\], as well as Sybil-resilient systems such as Whanau \[65\]. Social networks are also used to build privacy-preserving applications: secure lookup services such as X-Vine \[75\] and private data sharing systems such as OneSwarm \[50\].

1.3 Research Questions

This thesis aims to address the following questions:

What is an effective and scalable approach to recommend content items in large OSM? Recommending interesting and relevant content from the vast repositories of OSM such as YouTube, Flickr and Digg is a significant challenge. Part of this challenge stems
from the fact that classical collaborative filtering techniques (such as k-Nearest Neighbor) cannot be assumed to perform as well in OSM as in much smaller systems (e.g., IMDb and MovieLens). Such techniques have severe limitations regarding data sparsity and scalability that are unfitting for OSM. Hence, it is vital to design scalable algorithms that can recommend content of interest and relevance.

**What is an effective defense strategy for a global reputation system under a targeted attack?** EigenTrust [57] is a renowned algorithm for reputation management. It incorporates the opinions of all users in the system to compute a global trust score for each user based on her past behavior, and relies on a set of pre-trusted users to guarantee that malicious users cannot subvert the system. Since EigenTrust is a global ranking algorithm, a successful targeted attack can have a system-wide negative impact. Hence, it is important to first identify the bottlenecks of the algorithm particularly during a targeted attack, and then address these shortcomings.

**How can the number of votes from Sybil identities on a content item be minimized?** Due to open membership access, voting on content items in OSM is susceptible to Sybil attacks. Malicious attackers can create multiple Sybil identities to outvote the real users of the system, and thereby usually promoting spam content. Although the state-of-the-art approach – SumUp [92] – limits the number of votes from Sybil identities to the number of attack edges with a high probability, we contend the resulting solution still leaves room for considerable damage by attackers. For instance, even if malicious users, either acting individually or colluding with each other, constitute a small fraction (e.g., 1%) in a large network of 1 million users, SumUp would face more than 10,000 attack edges as well as equivalent number of votes from Sybils. Such a scale of the attack even with conservative estimates motivates the need for an approach to further reduce the votes from Sybils.

**Can the effectiveness of a social network-based Sybil detection scheme be improved by incorporating additional information?** Sybil detection schemes such as Sybil-Limit [104], SybilInfer [40] and SybilRank [29] leverage only the social network structure to differentiate honest and Sybil nodes by essentially computing rankings for each node in the network [97]. The effectiveness of each scheme depends on its ability to rank honest nodes above Sybil ones with a high probability. However, all these schemes exhibit two major vulnerabilities: (i) their effectiveness decreases with the increase in attack edges, and (ii) the more targeted the attack edges, the worse their effectiveness. To improve the effectiveness of these schemes, it is worthwhile to explore and exploit additional information in the system on top of the social network [103].

**Are the design considerations of the current social network-based distributed systems reasonable?** Recent years have seen many research initiatives to build robust and secure distributed systems using social networks. Primary examples of these systems are decentralized Sybil defenses such as SybilGuard [105], SybilLimit [104], and Gatekeeper [91].
Sybil-resilient systems such as Whanau [65], and privacy-preserving applications such as X-Vine [75] and OneSwarm [50]. However, all these distributed systems have been built and evaluated on centralized OSNs. It requires a great leap of faith in these studies to assume that a distributed system’s overlay network bootstrapped from an OSN has properties similar to its centralized counterpart. Designing a practical social network-based distributed system requires the understanding of the characteristics of operational distributed social networks.

1.4 Contributions and Thesis Outline

The contributions of this thesis are as follows:

**A link prediction approach to recommendations in large OSM (Chapter 2)** We employ adaptations of popular Link Prediction algorithms [67] that were shown to be effective and scalable in massive online social networks [89] for recommending items in OSM. We evaluate these algorithms on a large dataset we collect from Flickr. Our results suggest that Link Prediction algorithms are a more scalable and accurate alternative to classical collaborative filtering in the context of OSM. Moreover, our experiments show that the algorithms considering the immediate neighborhood of users in a user-item graph to recommend items outperform the algorithms that use the entire graph structure for the same. Finally, we find that, contrary to intuition, exploiting explicit social links among users in the recommendation algorithms improves only marginally their performance. This chapter is largely based on our work published in [34].

**Personalizing EigenTrust in the face of centrality attack (Chapter 3)** Our analysis reveals that EigenTrust (ET) is vulnerable to community structure and a novel targeted attack based on eigenvector centrality, since ET ranks nodes close to the pre-trusted ones higher than those further away. To address these shortcomings, we propose Personalized EigenTrust (PET) which (i) enables each user to choose her own trusted users from the social network, thereby eliminating the need of pre-trusted users and making the system self-sufficient, (ii) is effective in systems operating under various transaction models based on distributions such as random, community-like and power-law, and (iii) is robust to many types of attacks including the targeted one based on eigenvector centrality. Our simulation results reveal that PET outperforms ET under diverse transaction models and attack strategies. This chapter is largely based on our work published in [33].

**Leveraging trust and distrust for Sybil-tolerant voting in OSM (Chapter 4)** We propose a mechanism to minimize the votes from Sybil identities by leveraging (i) trust which is inherent in the social network among users in OSM, and (ii) distrust between honest users, who identify some of the spam content items, and the Sybil identities who promoted them. Modeling trust and distrust in the system as a signed network, our method proceeds in two phases. First, we identify nodes and edges that constrain paths along positive edges
between the endpoints of each negative edge. Second, we limit the votes from Sybil voters whose paths to honest nodes pass across these bottlenecks. Our simulation results on datasets of popular OSM show both the feasibility of incorporating distrust alongside trust to defend against Sybil attacks, and that our method outperforms the state-of-the-art approach, SumUp. This chapter is largely based on our work published in [35].

**Incorporating distrust into a Sybil detection scheme (Chapter 5)** Inspired by the previous chapter, we adopt a similar approach and rationale to a more generic problem of Sybil detection in OSNs. First, we build a resistance network on top of the signed network such that each positive edge over which a path corresponding to a negative edge passes is annotated with the endpoints of the negative edge. Such annotations add accountability as to who initiated a negative edge. We also limit the number of such paths passing over a positive edge. This bounds the counter attack capability of the adversary who may want to initiate negative edges from Sybil nodes to honest nodes. Moreover, it addresses some of the limitations of the design of the resistance network in our previous chapter. Second, we adapt SybilRank [29] to incorporate the resistance network to distribute trust in the network from a few seed nodes. Negative edges from honest nodes to Sybil nodes reduce the amount of trust to flow from the honest region to the Sybil region, enabling our method to differentiate honest and Sybil nodes with a high probability. Our experiments on popular OSN datasets show that our method significantly outperforms SybilRank even under targeted attacks. This chapter is largely based on our work which is under submission.

**Social networks meet distributed systems (Chapter 6)** With a design goal of building social network-based distributed systems, we study various properties of a Yahoo! Messenger which is a distributed social network in the wild used by millions of users. Our analysis reveals that Yahoo’s social network graph is fast mixing, and the network exhibits heavy churn. Such churn disintegrates the trust overlay into multiple disconnected components. We show that 2-hop neighborhood of each user can be leveraged to improve the network connectivity. A consequence of exploiting all the 2-hop neighbors is an order of magnitude of increase in the attack capability of an adversary. We explore a new design point in this tradeoff between network connectivity and attack resilience. We propose an adaptive 2-hop design that improves the connectivity of nodes as well as their resilience to attacks by forming relationships with only select 2-hop neighbors. Our trace-driven experiments under trying conditions reveal that our adaptive 2-hop method fills an important point in this design space making it significantly more resilient to churn than one-hop solution at the cost of modest increase in the attack capability of the adversary. This chapter is largely based on our work which is under review.

**Conclusions and future work (Chapter 7)** We summarize our main conclusions from the thesis, and throw light on the road ahead.
Chapter 2

A Link Prediction Approach to Recommendations in Large OSM

Online Social Media (OSM) such as YouTube, Flickr, and Digg have transformed the way media is shared and consumed. Prior to OSM, content distribution to large audiences was costly enough to have its control lying in a few hands: typically, those of professional and commercial producers. However, with technological advances in the recent years, the cost of producing and distributing media has drastically reduced. OSM have created venues where many of those who were previously passive content consumers now publish and share videos, photos, news articles and other content.

The democratic nature of publishing and sharing through OSM has attracted millions of users to contribute to these systems. Such large user bases, combined with the low cost and effort required to produce and publish content, typically create massive and fast-growing content repositories in OSM [30]. On the one hand, such sheer volume of content may cater to varied tastes of users. On the other hand, this scale gives rise to the problem of information overload, also formulated as the Babel objection [21]: differentiating quality from noise in such large user-generated data is very difficult. In essence, finding interesting and relevant content in OSM is a significant challenge.

Recommendation algorithms are a candidate solution to address this problem. OSM can use these algorithms to suggest interesting and relevant content to users based on their past preferences, thereby partially automating the process of content discovery. In spite of this potential, however, few studies have explored recommending items in the context of OSM (with the exception of YouTube [19]). Collaborative filtering (CF) techniques have been established as a good fit for this purpose in editorially-generated content systems (Non-OSM), such as MovieLens and Netflix, but there is little or no evidence that these techniques would perform well if applied to OSM.

Motivation and Challenges. There are primarily two reasons why classical CF techniques
which were found to be effective for Non-OSM – such as k-Nearest Neighbor – cannot be assumed to perform as well in OSM. First, the user-item matrix is substantially larger and sparser in OSM than in Non-OSM. This difference exacerbates the already existing limitations in CF techniques regarding scalability and sparsity [68]. While Non-OSM such as Internet Movie Database have about a million items [30], OSM may contain hundreds of millions of items. For instance, YouTube has recently claimed [14] that hundreds of thousands of videos are uploaded everyday on its website, accounting for 24 hours of videos every minute. Also, Flickr has celebrated the five billionth photo [1] uploaded in September 2010, and 3000 photos are being uploaded every minute.

The second reason that challenges the application of classical CF techniques in OSM relates to the lack of editorial control in these systems. While all items in Non-OSM are introduced and organized in respective categories by a few trusted editors, OSM contain content published by millions of users, and not all items are properly categorized and/or of good quality. This essentially magnifies the problem of users to sift through noise and find content of interest and relevance.

**Approach and Contributions.** In this chapter, we employ adaptations of Link Prediction algorithms [67] for recommending items in large OSM. These are a family of graph-based algorithms from the social network analysis literature which were found to be effective in predicting new links that might form in a given social network graph. A recent study [89] has shown that these algorithms are effective in predicting the formation of links between users in large and sparse social network graphs such as those of YouTube, Flickr, Digg, and LiveJournal. Inspired by their effectiveness to handle graphs with millions of nodes, we adopt these algorithms for CF in OSM. To apply them, we modify these algorithms to suit user-item graphs, since they were originally designed for graphs with only one kind of nodes (e.g., users). We use six Link Prediction algorithms in this study based on node-neighborhood, popularity, and path-ensemble.

We evaluate the recommendation performance of these algorithms on a large dataset we collected from Flickr containing 120,812 users and 83,435 photos. We use the popular item-based collaborative filtering technique [68] as a baseline to analyze the effectiveness of Link Prediction-based recommendation.

Our results show that the adapted Link Prediction algorithms outperform a widely used item-based CF technique [68]. Moreover, the relative performance of different Link Prediction algorithms unveils that most users are interested in photos within a short distance from them in the user-item graph. We also examine whether exploiting the explicit relationships among users which are often present in OSM improves the recommendation performance of our algorithms. Our findings suggest that, contrary to intuition, there is only a slight improvement in recommendation performance. Finally, for all Link Prediction algorithms, the more friends a user has and the more photos she bookmarks, the better the recommendations
Chapter outline. Section 2.1 characterizes OSM and then presents related work. Section 2.2 describes how Link Prediction algorithms can be used for collaborative filtering in OSM. The experimental methodology adopted for this chapter is discussed in Sections 2.3. We evaluate these algorithms under various scenarios in Section 2.4. The chapter is concluded in Section 2.5.

2.1 Background and Related Work

In this section, we first characterize what constitutes OSM and then review the literature related to recommendations and social influence in OSM.

2.1.1 Online Social Media

Two main characteristics define the current Online Social Media (OSM). First, any user can publish and share content items that other users can view and express opinions about, in the form of ratings, bookmarks, and comments. For example, in YouTube, a user can upload a video which other users can view, give a rating (Like/Dislike), or add as a favorite. Similarly, in Flickr, a user can upload a photo which other users can view or add as a favorite. Digg has similar features as well: a user can submit a story which others can view or bookmark as a digg. Additionally, users can comment on any content item in each of these OSM.

Second, in OSM, users can form social relationships with other users. These are usually framed as friendships or subscriptions, and primarily indicate interest of a user in another user’s activity. For instance, in YouTube, a user can subscribe to other users to keep updated of their uploaded videos as well as browse through the videos they have added as favorites. Flickr also allows a user to add others as contacts, which helps her to keep abreast of their activity. A Digg user can also add others as friends to follow their submissions and diggs. A relationship between any two users in such systems is typically asymmetric, i.e., a friendship link from a user A to user B means that the former is interested in the latter’s activity, but not necessarily vice-versa.

2.1.2 Item Recommendations in OSM.

Few studies have explored item recommendations in OSM. As of 2010, the work on YouTube by Baluja et al. [19] was the only extensive study on recommending items in OSM. The authors propose a novel graph-based technique called Adsorption, which recommends videos given a co-view graph representing which user viewed what video. Adsorption considers a video is relevant to a user if there are many short paths in the co-view graph be-
tween the user and the video which avoid high-degree nodes. This method is similar to Katz Measure [59], one of the Link Prediction algorithms we use in this chapter.

2.1.3 Social Influence in OSM.

Some recent studies analyzed the role of social (user-user) links for disseminating and promoting user-generated content in OSM. For example, an in-depth analysis of dissemination of photos along user-user links in the Flickr social network revealed that over half of all favorite-markings are exchanged between friends, thereby indicating a significant social influence on this behavior [31]. Another extensive study on diffusion of user-generated content in YouTube [80] found that social influence plays a prominent role in both the success of video as well as the magnitude of the impact. Different from these studies which analyze content diffusion along social links, our work focuses on content adoption by users irrespective of social influence.

2.2 A Link Prediction Approach to Collaborative Filtering

The fundamental task of collaborative filtering (CF) is to predict the interestingness and relevance of an item to a user. This is typically done based on how closely this item is related to the user’s tastes. Basically, proximity – the measure of closeness – lies at the heart of CF. The challenge of applying CF to OSM translates into developing methods for calculating proximity that are both effective and scalable for large user-item spaces.

In this chapter, we advance the hypothesis that the methods based on Link Prediction algorithms [67] provide an effective and scalable solution for CF in OSM. Like CF, the underlying rationale of most Link Prediction algorithms is based on proximity. The Link Prediction problem is to predict the formation of links in a social network graph, and the corresponding solutions explore the principle that the closer two nodes are in such a graph, the higher the chance a link between them forms. Unlike classical CF techniques, however, some of the Link Prediction algorithms have been shown [89] to be highly scalable, performing well in massive and sparse social network graphs such as those of YouTube, Flickr, Digg, and LiveJournal.

To bridge Link Prediction algorithms and CF in OSM, we develop variants of some of the Link Prediction algorithms in the literature [67, 89] to suit user-item graphs. These variants are necessary because Link Prediction algorithms were mainly designed for graphs containing only one kind of nodes (e.g., users). In contrast, CF in OSM concerns with predicting links between two types of nodes: users and items. Hence, CF can essentially be viewed as a user-item link prediction problem: given a graph with users and items as nodes, and users’ tastes in items represented by user-item edges, how accurately can we infer whether a link will form between an item and a particular user? Each Link Prediction algorithm we use in
the chapter solves this problem by calculating a proximity score that expresses how relevant any item is to a particular user in the graph.

2.2.1 Notation

We model a user-generated content system as a directed graph $G = (V, E)$ where the set of nodes $V$ consists of all users $U$ and items $I$ present in the system ($V = U \cup I$), and $E$ is the set of edges that represent various relationships among these nodes ($E \subseteq U \times U \cup U \times I$). The relationship between a user node and an item node is represented by two edges in opposite directions, which are referred to as user-item links. The relationship between two users $p$ and $q$ is represented by either a single directed edge from $p$ to $q$ if $p$ follows $q$, or two directed edges in opposite directions if both $p$ and $q$ follow each other. Such a relationship is referred to as user-user link. However, there exists no relationship between two item nodes.

The adjacency matrix $A$ of the user-item graph $G$ is such that $A_{x,y} = 1$ if edge $(x,y) \in E$, otherwise $A_{x,y} = 0$. In addition, we define $N_u(x)$ and $N_i(x)$ as the set of users and items that are neighbors of a node $x$ in the user-item graph, respectively. That is, $N_u(x) = \{ y \mid (x,y) \in E \text{ and } y \in U \}$ and $N_i(x) = \{ y \mid (x,y) \in E \text{ and } y \in I \}$. Finally, we denote $N^{-1}(x)$ as the set of nodes having $x$ as their neighbor ($N^{-1}(x) = \{ y \mid (y,x) \in E \}$).

2.2.2 Algorithms

We adapt and employ six Link Prediction algorithms in this chapter. Link Prediction algorithms can be broadly classified according to the node characteristic they rely on when calculating proximity. Most of these algorithms are based on node neighborhood, popularity, or path ensemble. The six algorithms we adapt include two algorithms based on each of these characteristics, namely:

- **Node-neighborhood-based**: Common Neighbors and Adamic/Adar;
- **Popularity-based**: Global Popularity and PageRank;
- **Path-ensemble-based**: Katz Measure and Rooted PageRank.

Among these, node-neighborhood-based algorithms have restricted scalability, and do not necessarily constitute a viable approach for OSM. However, we adapt and include them in this study because such methods were shown to provide highly effective predictions from both theoretical \[86\] and practical perspectives \[67\], even for large datasets \[89\]. As such, they are used in our experiments as references for the performance that should ideally be achieved by the other two groups of algorithms, which have been shown to scale for graphs with millions of nodes \[89\].
In the rest of the section, we propose these algorithms for a bipartite user-item graph which does not include edges among users. Later, in Sec. 2.4.2, we revisit these algorithms taking into account user-user links as well. Each algorithm $ALG$ calculates a proximity score $ALG(u, z)$ for a given user $u$ and item $z$.

**Common Neighbors.** The rationale behind this algorithm is that the greater the intersection of the neighbor sets of any two nodes, the greater the chance of future association between them \[67\]:
\[
CN(u, z) = \sum_{v \in N_i(u)} |N_u(v) \cap N_u(z)|.
\]

**Adamic/Adar.** This method also measures the intersection of neighbor-sets of two nodes in the graph, but emphasizes the smaller overlap \[17\]:
\[
AA(u, z) = \sum_{j \in N_i(u)} \sum_{v \in N_u(z) \cap N_u(j)} (\log |N_i(v)|)^{-1}.
\]

**Global Popularity.** This method is a variant of preferential attachment, which quantifies the popularity of a node as its degree, and uses this popularity as the proximity from any other node \[67\]:
\[
GP(u, z) = N_u(z).
\]

**PageRank.** This algorithm leverages the link structure of a graph to quantify the popularity of each node \[81\], by assigning to this node a global rank $PR(u, z) = PR(z)$, where
\[
PR(z) = (1 - d)(|V|)^{-1} + d \sum_{x \in \mathcal{N}^{-1}(z)} PR(x)(| \mathcal{N}(z) |)^{-1},
\]
and $d$ (set to 0.85 in this chapter) is the teleportation parameter.

**Katz Measure.** The proximity score for this method is calculated by considering all paths in the graph \[59\]. The logic is that the more the paths between any two nodes in the graph and the shorter these paths, the greater the “bond” between these nodes: \[59\]
\[
KM = (I - \beta_K A)^{-1}
\]
where $\beta_K$ is the damping factor.

**Rooted PageRank.** This method \[67, 89\] is a variant of the personalized PageRank algorithm. It attempts to capture the probability of random walks starting from two nodes in the graph to come across each other, and uses this probability to quantify the proximity between these nodes. Let $D$ be the diagonal matrix with $D[i, i] = \sum_j A[i, j]$, then
\[
RPR = (1 - \beta_{RPR})(I - \beta_{RPR}D^{-1}A)^{-1},
\]
where $\beta_{RPR}$ is the teleportation parameter.

In each of the matrices $KM$ and $RPR$, the element in row-$u$ and column-$z$ denotes the proximity score for the respective algorithm, given a user $u$ and an item $z$. For scalable computation of the matrix inversion used in Katz Measure and Rooted PageRank, we use a dimensionality reduction technique called Proximity Embedding \[89\]. We also use $\beta_K = \beta_{RPR} = \beta = 0.005$. 

Table 2.1: Summary of Flickr dataset

<table>
<thead>
<tr>
<th></th>
<th>Jan 1, 2010 to Jan 31, 2010</th>
<th>Feb 1, 2010 to Mar 31, 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training period</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing period</td>
<td></td>
<td></td>
</tr>
<tr>
<td># Users active in both periods</td>
<td>120,812</td>
<td></td>
</tr>
<tr>
<td># Photos active in both periods</td>
<td>83,435</td>
<td></td>
</tr>
<tr>
<td># Favorite markings in training period</td>
<td>1,755,575</td>
<td></td>
</tr>
<tr>
<td># Favorite markings in testing period</td>
<td>1,234,854</td>
<td></td>
</tr>
<tr>
<td>Median favorites per user</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Median contacts per user</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

2.3 Experimental Methodology

2.3.1 Dataset

We evaluate our algorithms on Flickr [4], which contains explicit user-item and user-user links, depicting tastes and sources of influence for any user respectively. With reference to the terminology introduced in Section 2.1 in Flickr, we consider *favorites* as user-item links\(^1\) and *contacts* as (directed) user-user links. Nevertheless, in the rest of the chapter, we use the terms ‘photo’ and ‘item’ interchangeably. Also, we intermingle the terms ‘contacts’ and ‘friends’.

2.3.2 Data Collection.

Flickr currently has several millions of users and items. Since obtaining all relevant data for this study is infeasible, we sample the user-item graph. We use snowball sampling [47], a popular approach in online social networks. This method leads to nearly complete data for a particular neighborhood of the graph, which is of interest to recommendation algorithms. The time window considered for collecting data about users and favorites is three months.

2.3.3 Dataset Description.

Table 2.1 presents the summary of the collected data. The data is divided into a training period of one month and a testing period of two months. Note that the median value for the number of favorites is five compared to 50 for contacts: the number of user-user links is nearly an order of magnitude larger than the number of favorite markings.

\(^1\) The data about who viewed what item is private to Flickr. We therefore limit our discussion to the data publicly available through the Flickr API [6].
2.3.4 Metrics.

The recommendation performance of the six Link Prediction algorithms is measured in our experiments in terms of precision, recall, and mean average precision (MAP). Precision measures the fraction of retrieved photos that are relevant, while recall measures the fraction of relevant photos that are retrieved. MAP is a single-figure measure that captures quality of precision across recall values.

To calculate these metrics, we adopt the approach of Baluja et al. [19]. Specifically, for each user $u$, we use the photos she bookmarked during the training period to generate a ranked list $R_u$ of photos for each given algorithm. The top-$N$ (in Sec. 2.4, $N = 200$) photos from this list are then used for evaluation. Let $R^i_u$ be the set of the first $i$ photos in $R_u$, and let $W_u$ be the set of photos a user $u$ bookmarked during testing period. Then, for $i < |R|$, for the user $u$ at rank-position $i$, we define: (i) precision: $p_u^i = |W_u \cap R^i_u|/|R^i_u|$ and (ii) recall: $r_u^i = |W_u \cap R^i_u|/|W_u|$. The computation of precision and recall at rank-position $i$ for each algorithm averages $p_u^i$ and $r_u^i$ across all users, respectively. The calculation of MAP for user $u$ averages precision values at rank-positions in the ranked list which match relevant items. MAP for each algorithm $ALG$ is calculated by averaging across all users.

These metrics reflect the effectiveness of the algorithms, but not their efficiency. The efficiency, and hence the scalability of these methods, has been extensively studied before [89], with results that also hold for user-item graphs.

2.3.5 Baseline Method.

We compare the recommendation performance of our algorithms against the widely-used item-based collaborative filtering technique [68] as a baseline. Although this method is infeasible for millions of users and items, it can still be applied to our dataset.

2.4 Evaluation

In this section, we evaluate the recommendation performance of our Link Prediction algorithms on the crawled Flickr dataset. We aim to answer the following questions:

(i) How do these algorithms perform in comparison to an item-based collaborative filtering technique [68], which we use as a baseline?

(ii) Does exploiting the knowledge of explicit user-user links improve recommendation performance?

(iii) How does the behavior of a user influence the quality of her recommendations?
2.4.1 Link Prediction Performance

In this experiment, we compare the recommendation performance of our Link Prediction algorithms against the item-based collaborative filtering technique as a baseline, considering the user-item graph with only user-item links. Table 2.2 shows the recommendation performance in terms of MAP, precision, and recall of each algorithm at various rank positions.

Overall, Link Prediction algorithms outperform the baseline item-based CF technique. Among Link Prediction algorithms, the algorithms based on node-neighborhood perform better than those based on popularity and path-ensemble. On the other hand, popularity-based algorithms perform the worst.

Both neighborhood-based algorithms – Common Neighbors and Adamic/Adar – have similar recommendation performance. Although Adamic/Adar has a marginally higher overall performance than that of Common Neighbors, the latter performs better for the top-few ranked items. A closer look at our datasets reveals that these algorithms perform well because most favorites marked by a node are in its close neighborhood. In our data, 75% of the favorites marked during the testing period were within a distance of three hops from the users who bookmarked them in the graph.

The recommendation performances of both popularity-based algorithms – Global Popularity and PageRank – are also similar. To examine why this happens, we compare the sheer popularity (node degrees) and PageRank values for each item. This analysis shows that the most popular items also have high PageRank values. We also note that recall values for these algorithms do not improve to the extent of other algorithms, suggesting that only the top few ranked items were of interest and relevance to users.

Among the path-ensemble algorithms, Katz Measure performs better than Rooted PageRank in terms of MAP, as well as precision for top-few ranked items. On the other hand, Rooted PageRank surprisingly has the highest recall value at 200-th rank-position among all algorithms. This shows that, although Rooted PageRank is able to correctly recommend more items than any other algorithm, the order in which these recommended items are ranked is not as effective as that of neighborhood-based algorithms or Katz Measure. We also note that varying the parameter of path-ensemble algorithms from $\beta = 0.005$ to an order of magnitude greater and smaller caused no performance changes.

We now focus on the performance of these path-ensemble algorithms while recommending items beyond three hops from users in the user-item graph. In this context, Rooted PageRank outperforms Katz Measure. The former has 1.5 times better precision at the top-ranked position and 5 times higher Precision@10 compared to Katz Measure. We also observe that Rooted PageRank has 6 times better recall for top-200 items than that of Katz Measure. These observations suggest that predicting more items which are beyond three hops correctly may be one of the reasons why the recall of Rooted PageRank improves significantly.
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>MAP</th>
<th>Precision@1</th>
<th>Precision@10</th>
<th>Precision@100</th>
<th>Recall@10</th>
<th>Recall@100</th>
<th>Recall@200</th>
<th>Item-CF</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN</td>
<td>6.434</td>
<td>7.130</td>
<td>1.730</td>
<td>0.577</td>
<td>2.298</td>
<td>5.732</td>
<td>7.468</td>
<td>1.914</td>
</tr>
<tr>
<td>AA</td>
<td>6.240</td>
<td>6.795</td>
<td>1.783</td>
<td>0.613</td>
<td>2.425</td>
<td>6.448</td>
<td>8.385</td>
<td>2.174</td>
</tr>
<tr>
<td>GP</td>
<td>6.566</td>
<td>7.269</td>
<td>0.817</td>
<td>0.198</td>
<td>1.120</td>
<td>2.185</td>
<td>2.810</td>
<td>2.874</td>
</tr>
<tr>
<td>PR</td>
<td>6.576</td>
<td>7.269</td>
<td>0.823</td>
<td>0.192</td>
<td>1.132</td>
<td>2.246</td>
<td>3.037</td>
<td>3.183</td>
</tr>
<tr>
<td>KM</td>
<td>5.783</td>
<td>6.898</td>
<td>1.365</td>
<td>0.547</td>
<td>1.713</td>
<td>5.583</td>
<td>7.754</td>
<td>7.185</td>
</tr>
<tr>
<td>RPR</td>
<td>4.561</td>
<td>5.061</td>
<td>1.308</td>
<td>0.609</td>
<td>1.571</td>
<td>6.238</td>
<td>9.198</td>
<td>8.385</td>
</tr>
<tr>
<td>Item-CF</td>
<td>5.183</td>
<td>5.702</td>
<td>1.529</td>
<td>0.525</td>
<td>2.045</td>
<td>5.762</td>
<td>7.914</td>
<td>7.468</td>
</tr>
</tbody>
</table>

Table 2.2: Recommendation performance (%) of the algorithms.
We observe that the overall values of MAP, precision, and recall in our experiments are considerably smaller than those in most studies in the field of information retrieval. This can be due to two reasons. First, the large size and sparsity of the dataset challenge the algorithms. We note here that the study on recommending videos in YouTube also reported low precision and recall values [19], suggesting that collaborative filtering in OSM is an especially difficult task. Second, some items that were recommended by these algorithms might have been bookmarked either before the training period or after the testing period, which the collected data does not capture.

2.4.2 Influence of Explicit Social Links

Our second experiment investigates whether exploiting explicit user-user links improves the recommendation performance of our algorithms. For this purpose, we first include both user-item and user-user links in the user-item graph and then employ adapted versions of our algorithms. Furthermore, we examine whether the direction of user-user links affects recommendations, by comparing the results in the original graph with those from a symmetrized graph. In the symmetrization, for each node pair containing only one edge, we add a link in the reverse direction.

The algorithms are adapted for considering user-user links as follows. The scores for node-neighborhood methods are redefined as

\[ CN(u, z) = |N_u(u) \cap N_u(z)| \]

and

\[ AA(u, z) = \sum_{v \in N_u(u) \cap N_u(z)} \frac{1}{\log|N(v) + N_u(v)|} \].

For Katz Measure and Rooted PageRank, we incorporate edges among users in the adjacency matrix of the graph before computing the scores using the same techniques. Since the popularity-based algorithms are unaffected by the user-user links in the graph, they are not considered in this experiment.

Figure 2.1 shows the recommendation performance of each algorithm across three scenarios: (i) the one with only user-item links (as a baseline), (ii) the one with the unmodified user-item graph considering user-user links and (iii) the one with reciprocated user-user links. We make the following observations:

Effect of User-User Links.

The overall recommendation performance of Common Neighbors, Adamic/Adar and Katz Measure slightly improves when user-user links are considered. Rooted PageRank, however, has a reduced performance in terms Precision@10. Although Katz Measure has a better Precision@1 than any other algorithm, its overall performance is still below that of neighborhood algorithms without considering user-user links. The precision of Rooted PageRank is reduced by 40%, which may be attributed to the increment in the node-degree for each user in the graph due to the inclusion of user-user links. This increment may reduce the influence of each neighbor during a random walk.
Effect of Link Symmetry.

The reciprocation of user-user links marginally improves recommendation performance. The low effect size indicates that the influence among users in Flickr is not usually mutual, i.e., if a user A has added a user B as a contact, and there is no reciprocation from user B, the influence of A’s tastes on B is small. Song et al. [89] had similar findings for the prediction of user-user links in Flickr: accuracy was nearly the same when graph was symmetrized.

2.4.3 Influence of User Behavior

We now examine how the behavior of a user influences the quality of her recommendations. In this experiment, user behavior is characterized by the number of items bookmarked and friends. The former indicates a user’s activeness, while the latter suggests the amount of social influence on her.

Effect of User Activeness.

We characterize users with different activity levels by grouping users in bins according to their activity. This is done approximating the number of bookmarked items of each user to the nearest power of 10. After this, the MAP value for users in each bin is averaged.

The left half of Figure 2.2 presents the recommendation performance of each algorithm as a function of the number of bookmarked items (favorites). The recommendation performance of each algorithm is in general higher for users that were more active than for those who bookmarked fewer items during the training period. A positive correlation between available information and recommendation quality is in accordance with the intuition in recommender systems. Nevertheless, these results show that this intuition is valid also in the context of OSM, and provide evidence on precisely how recommendation performance
Figure 2.2: Recommendation performance with respect to user behavior.

The right half of Figure 2.2 presents the recommendation performance of each algorithm as a function of users’ number of friends. The recommendation performance of each algorithm is in general higher for users with a large number of contacts compared to those with few. MAP for users with around 1000 friends is 2-3 times higher compared to those with fewer than 100. It is also interesting to note that the recommendation performance of each algorithm does not vary much for users with fewer than 100 friends, but it increases gradually until 1000. This pattern is similar to that of favorites’, except that the latter’s improvement starts at 10 favorites. This may be ascribed to the ratio of median values of friends and favorites per user, which is precisely 10 for this dataset. In spite of this difference, however, we may conclude that the more friends a user has and the more items she bookmarks, the better the recommendations to her.

2.5 Conclusions

In this chapter, we advanced a Link Prediction-based approach for recommending items in large-scale OSM. We evaluated the recommendation performance of six Link Prediction algorithms on a large dataset we collected from Flickr, with the widely-used item-based
collaborative filtering technique as a baseline. Three of the Link Prediction algorithms – Common Neighbors, Adamic/Adar and Katz Measure – performed consistently better than the item-based collaborative filtering technique. We found that neighborhood-based methods outperform all other algorithms, suggesting that users are mostly interested within a small proximity of their tastes in the user-item space. Rooted PageRank, on the other hand, was very effective in recommending items that are beyond three hops from users in the user-item graph. In addition, exploiting the explicit relationships among users improved recommendation performance only marginally, contrary to our expectations. With respect to user behavior, the larger the number of friends of a user or the number of photos bookmarked by her, the better the quality of recommendations to her, implying that social influence and activeness of user does affect the performance. Finally, the low values of precision and recall in our study, along with the only other extensive investigation in our knowledge, suggest that recommending items in OSM is highly challenging, and hence requires significant attention from IR research community.
Chapter 3

Personalizing EigenTrust in the face of Centrality Attack

Defending against malicious behavior in open systems is a longstanding challenge. A family of random walk-based trust schemes have been proposed over the last decade to counter adversaries in various fields such as web search [49], P2P reputation systems [57][107], and Sybil defenses [40][104][105]. Despite considerable differences, we notice that each of these schemes employs a fundamental principle – random walk with restart from trusted nodes – to rank nodes in the system in order to differentiate honest nodes from malicious ones. A recent study [97] shows that Sybil defense schemes [40][104][105], a subset in this family, are highly vulnerable to community structure and targeted attacks. In light of this study, we hypothesize that these concerns may affect other schemes in this family. In this chapter, we revisit a renowned trust scheme in this family – EigenTrust [57] – to verify our hypothesis.

EigenTrust (ET) is an algorithm for reputation management in adversarial P2P systems [57]. It calculates a ‘global’ trust score for each peer based on its past behavior by incorporating the opinions of all peers in the system. ET relies on a set of pre-trusted peers (usually chosen by the system designers) to guarantee that adversaries cannot subvert the system.

Our analysis of ET reveals that peers that are close and better connected to the pre-trusted peers are ranked higher (based on their trust scores) than the rest of the network. This insight has two main implications. First, ET can essentially be viewed as a community detection algorithm which identifies communities around pre-trusted peers. As a consequence, peers in communities that are ‘far away’ from pre-trusted ones would be ranked very low despite potentially being honest.

Second, our insight motivates malicious peers to employ a novel attack strategy based on eigenvector centrality [23]. That is, malicious peers improve their ranking by behaving well with top ranked ones, and acting poorly with low ranked peers with little consequence. Since ET is a global ranking algorithm, such a targeted attack has system-wide negative
impact. Both these implications highlight two main bottlenecks of ET: pre-trusted peers and global ranking.

To address these shortcomings, we propose Personalized EigenTrust (PET) which enables each peer to (i) choose its own set of trusted peers from the social network of peers, and (ii) calculate personalized rankings of other peers. The use of social network-based trusted peers simply eliminates the need for the pre-trusted ones, thereby making the system autonomous in that there is no ‘central’ element such as pre-trusted peers. As a result, there is no ‘single point of attack’ that adversaries can predominantly target.

We evaluate the performance of PET under diverse conditions through simulations. We reach the following conclusions. First, PET outperforms ET (i) under various transaction models based on distributions such as random, community-like, and power-law, and (ii) in the face of different attacks such as collusion with spies, centrality-, and traitor-based. Secondly, the eigenvector centrality attack is shown to be more devastating than any of the attacks previously studied [57]. ET performs worse than a random technique when this strategic attack is performed. Third, ET performs particularly bad under community-based transaction models. Fourth, PET is very effective in ranking honest peers higher than malicious ones. This shows that, when presenting the user the search results, good hits are shown at the top while the bad ones are pushed to the bottom of the ranked list. Lastly, but equally as important, social network-based trusted peers in PET address the cold start problem of newcomers during their bootstrap phase into the system. This potentially improves user experience which is vital in retaining the newcomers in the system.

The insights from our study have important implications for existing and future designs of flow-based trust schemes in adversarial systems. First, such trust schemes need to be evaluated under diverse network topologies (transaction models) to better understand their strengths and weaknesses. Second, trust schemes based on (variants of) pagerank or max-flow need to be evaluated under attacks based on eigenvector- or betweenness centrality, respectively.

Our study is significantly different from previous efforts at personalization of EigenTrust [36, 37, 52]. At a high level, their main premise is that the pre-trusted peers could either leave the system or be compromised, thereby weakening the system. Although their concerns are valid, our study shows that, even in a churn-free and non-compromised system, ET fares poorly when the network exhibits community structure and/or centrality attacks are performed.

3.1 Analysis of EigenTrust

Our goal in this section is to better understand the process by which EigenTrust [57] computes the rankings of peers.
3.1.1 Algorithmic overview of ET

Consider a P2P file-sharing system in which a peer $i$ after downloading a file from another peer $j$ rates the transaction as positive (+1) or negative (−1) based on whether the downloaded file was authentic or not. Peer $i$ then computes a local trust value $S_{ij}$ which represents its net number of authentic downloads from peer $j$. Next, peer $i$ computes a normalized local trust value $C_{ij} = \max(S_{ij}, 0)/\sum_j \max(S_{ij}, 0)$ representing the extent of trust $i$ has in $j$ ($C_{ij} \in [0, 1]$). Since a typical peer $i$ downloads from only a small subset of the population, its trust-based view $C_{i,*}$ of the network is limited.

To expand its knowledge, peer $i$ incorporates the opinions of its neighbors and the neighbors of its neighbors and so on to compute its reputation vector $\vec{r}_i = (C^T)^n C_{i,*}$ after $n$ iterations. For large $n$, each peer’s $\vec{r}_i$ converges to the same vector $\vec{r}$, the left principal eigenvector of $C$. In essence, $\vec{r}$ is considered the global reputation vector, and $\vec{r}[j]$ denotes the global reputation score of peer $j$.

To counter the gaming of the system, ET employs a set of pre-trusted peers which all peers (or at least the honest ones) trust such as the designers and early users, since they are less likely to disrupt the system. The opinions of these pre-trusted peers are weighed more than the others to compute the final global reputation vector $\vec{r}$, which is redefined as:

$$\vec{r} = \alpha C^T \vec{r} + (1 - \alpha) \vec{d}$$

(3.1)

where the teleportation parameter $\alpha \in [0, 1]$ is a constant, and $\vec{d}$ is the static distribution vector representing the set of pre-trusted peers. For every pre-trusted peer $i$, $\vec{d}[i] = 1/p$ where $p$ is the number of pre-trusted peers; otherwise, $\vec{d}[i] = 0$.

ET can also be interpreted in the context of Markov chains. Let the trust matrix $C$ be modeled as a weighted directed graph in which a peer represents a node and the element $C_{i,j}$ represents the weight of the edge from peer $i$ to peer $j$. Consider a random walker, starting from one of the pre-trusted peers, wandering the graph. When at a particular node $i$, this walker with a probability $\alpha$ selects a random neighbor $j$ based on $C_{i,j}$, and with a probability $1 - \alpha$ jumps to one of the pre-trusted peers. The more often this walker visits a node, the higher its reputation score (and hence, its ranking).

3.1.2 ET as a community detection algorithm

Since each random walk starts from one of the pre-trusted nodes, we hypothesize that nodes close to the pre-trusted nodes have a higher likelihood to be visited by the random walker, and hence, have higher reputation scores compared to farther nodes. To verify our hypothe-
sis, we adopt a similar methodology as [97]. Specifically, we first build a synthetic network using the Barabasi-Albert preferential attachment model [20] with 512 nodes and an initial node degree of 8, and then rewire it such that two densely connected communities ($G_1$ and $G_2$) of 256 nodes each are formed, connected by a small number of edges (Fig. 3.1(a)). Next, we randomly choose a node in one of the communities (say $G_1$) as the pre-trusted node. We then compute EigenTrust reputation scores for all nodes in the network, based on which we calculate their rankings.

Figure 3.1(b) illustrates the correlation between the rank of a node and its connectivity to the pre-trusted node. The horizontal axis denotes the size of the partition containing top-$k$ ranked nodes. For example, the value of $k = 10$ splits the ranking into two parts: one with the top-10 ranked nodes, and the other with the rest. The vertical-axis denotes the conductance of the partition containing top-$k$ ranked nodes. Conductance is a widely used metric to evaluate the quality of community structure [64] which measures how closely a subset of nodes are connected among themselves relative to the rest of the network. The values of conductance range from 0 to 1, with smaller values indicating stronger communities.

Figure 3.1(b) plots the conductance as we vary the size of the partition containing the top-$k$ ranked nodes. Most of the nodes in the same community ($G_1$) as the pre-trusted node are ranked higher than those in the other community ($G_2$). This is because the former are better connected to the pre-trusted nodes than the latter. There is also a strong community structure for the partition of nearly half the nodes in the network. However, adding more nodes from the bottom half ranked peers increases conductance, thereby decreasing the community structure for larger partitions.

Hence, ET can be viewed as a community detection algorithm which identifies communities around pre-trusted nodes.

### 3.1.3 Eigenvector centrality attack & its impact

ET is essentially a variation of eigenvector centrality [23] which is a measure of the relative importance of a node in a graph. The global reputation score, and thus the rank, of a peer in ET can be viewed as a measure of its eigenvector centrality with respect to pre-trusted peers. Rephrasing the finding in Sec. 3.1.2, nodes close to the pre-trusted peers are top (eigenvector) central peers, while the nodes further away are less central peers. Hence, a straightforward attack strategy for malicious peers is to form edges with top central peers in order to get close to the pre-trusted peers, thereby gaining high rank which they then exploit to act poorly with the rest of the peers (which are less central).

We now validate this attack strategy by demonstrating how an adversary can improve its rank in the system. We first construct a synthetic network using the Barabasi-Albert preferential attachment model [20] with 512 nodes and an initial degree of 8, and then we randomly choose a pre-trusted node. Next, we choose a node $i$ with an average rank (say
Figure 3.1: Ranking in EigenTrust. 3.1(a) A pictorial illustration of a synthetic network with two communities $G_1$ and $G_2$, with the pre-trusted node in $G_1$. 3.1(b) Conductance versus partitions based on node rank. Color represents the community to which the node at a particular rank $k$ belongs. Conductance shows the quality of community structure of top-{$k$} ranked nodes relative to the rest of the network. 3.1(c) New rank obtained by targeting a particular ranked node $k$. 
256), and add one edge to another target node \( j \).

Figure 3.1(c) plots the new rank of this node \( i \) versus the rank of the target node \( j \). Targeting top ranked nodes significantly improves \( i \)'s ranking. For instance, just a single edge with a top-5 ranked node increases \( i \)'s ranking from 256 to below 50. However, adding an edge to a lower ranked node has little effect on \( i \)'s rankings. This confirms that a dominant strategy for malicious nodes is to behave well with top central nodes, and to act maliciously with the rest with little consequence.

3.1.4 Limitations

The insights gained from Sec. 3.1.2 and Sec. 3.1.3 have negative implications on two key aspects of ET:

**Pre-trusted peers**

The coverage of random walks from pre-trusted peers becomes vital to the performance of ET. As the system grows, more and more communities typically evolve due to varied tastes in users. If the pre-trusted peers are selected poorly, nodes in far off communities would be ranked very low despite being honest.

**Global ranking**

Since ET is a global ranking mechanism, a centrality attack will have system-wide negative impact. For instance, strategic malicious peers can behave well with a few top ranked peers (minority), while behaving maliciously with the others (majority). In addition, the rating by a user is often subjective, and hence a general agreement is difficult to reach in all scenarios. Consider a typical example: a poor quality video may be disregarded by some users, but may suffice for others due to its content and significance.

In short, global consensus on pre-trusted nodes makes ET ineffective when either the network exhibits a community structure or when a centrality attack is applied.

3.2 Personalized EigenTrust

We argued in the previous section that the two main bottlenecks of ET are *pre-trusted peers* and *global ranking*. Here we describe our approach to address these shortcomings.

3.2.1 Approach

We propose Personalized EigenTrust (PET) which enables each peer to (i) choose its own set of trusted peers, instead of the pre-trusted ones as in ET, and (ii) formulate its personal-
ized view of the system, instead of a global one as in ET. More formally, a peer \( i \) computes its personalized reputation vector \( \vec{r}_i \) (similar to Eq. (3.1)) using its set of trusted peers (including itself) in the form of trust vector \( \vec{d}_i \):

\[
\vec{r}_i = \alpha C^T \vec{r}_i + (1 - \alpha) \vec{d}_i
\]  

(3.2)

We now discuss the interpretation of PET in two different contexts. From the centrality point of view, the personalized reputation score of a peer \( j \) from another peer \( i \)'s perspective is a measure of \( j \)'s eigenvector centrality with respect to \( i \)'s trusted peers (which includes itself). From a random walker’s perspective, a peer \( i \) wishing to evaluate other peers in the system begins a random walk from one of its trusted peers (including itself) and then wanders the system. With a probability \( 1 - \alpha \), this walker jumps back to one of its trusted peers and restarts the random walk. This is significant on two fronts. First, the walker traverses mostly the relevant part of the system which gives her a personalized view. Secondly, the walker remains close to its trusted nodes, and hence reduces the chances of escaping into the malicious region.

PET leverages the trust inherent in the social network among the peers. The use of social network-based trusted peers simply eliminates the need for the pre-trusted peers. Essentially, this makes the system autonomous in that there is no ‘central’ element such as pre-trusted peers that significantly influences reputation evaluation. As a result, there is no ‘single point-of-attack’ that adversaries can predominantly target. We discuss more details on the role of social networks in Section 3.5.

### 3.2.2 Basic computation

We now describe how each peer \( i \) computes its personalized reputation vector \( \vec{r}_i \). Peer \( i \) first gathers all columns of the matrix \( C \) which simply correspond to the respective peers. Next, \( i \) computes \( \vec{r}_i \) using its trusted peers in the form of \( \vec{d}_i \) (Eq. (3.2)). We note that the information of its trusted peers is kept locally. This ensures that the personalized reputation vector \( \vec{r}_i \) is private to \( i \). Here we assume that the computation of PET is oblivious to the underlying overlay network. That is, the matrix \( C \) can be gathered using gossiping in unstructured networks or DHTs in structured networks.

We now discuss the complexity of computing PET in this fashion. Suppose there are \( n \) peers in the network, and their transactions result in the matrix \( C \) having \( m = O(n) \) non-zero elements. Here, we assume that \( C \) is sparse, i.e., \( m \ll n^2 \). For each peer \( i \), computing its personalized reputation vector \( \vec{r}_i \) takes \( O(m) \) network overhead, \( O(m) \) storage space, and \( O(ml) \) time where \( l \) is the number of iterations for \( \vec{r}_i \) to converge. This implies that, for a network of 1 million nodes and 10 million non-zero floating point elements in \( C \), the storage required and the network overhead for each peer is around 40 MB. Considering the fact that the size of today’s popular files in P2P networks is in the order of 1 GB, downloading 40
MB required for reputation calculation seems reasonable since the goal in adversarial P2P file-sharing systems is to download an authentic version of a file the very first time. This improves user experience as well as reduces the wastage of resources in terms of time and bandwidth.

3.2.3 Scalable computation

We now describe how PET can be computed in a scalable fashion. Let $R$ be a matrix in which the $i$-th column represents the vector $\vec{r}_i (= R_{:,i})$. Also, let $D$ be defined similarly. Bringing all under one umbrella, Eq. (3.2) is reformulated as:

$$R = \alpha C^T R + (1 - \alpha) D$$  \hspace{1cm} (3.3)

Solving for $R$, Eq. (3.3) is rewritten as:

$$R = (1 - \alpha)(I - \alpha C^T)^{-1} D$$  \hspace{1cm} (3.4)

The right-hand side of Eq. (3.4) comprises two parts. The first part $(I - \alpha C^T)^{-1}$ takes into account the transaction behavior of peers in the system, while the second part $D$ represents the trust relationships in the social network. This allows us to clearly differentiate the computational aspects as well.

The matrix inverse $S = (I - \alpha C^T)^{-1}$ can be efficiently computed using matrix dimensionality reduction techniques such as proximity embedding [89]. This technique factorizes $S$ into two matrices $U$ and $V$ of smaller dimensions such that $S_{n \times n} = U_{n \times p} V_{p \times n}$ where $p << n$. We employ a super-peer approach to compute $U$ and $V$ by exploiting the heterogeneity among peers in the network in terms of resources such as computational power, memory and online time [32]. A fraction of powerful peers are elected as score managers which coordinate to compute $U$ and $V$ in a distributed fashion. Song et al. [89] have observed that the factorization of $S$ was nearly 15 times faster by using 150 peers compared to a single instance.

To compute $R_{:,i}$, peer $i$ first requests the vector $S_{:,j}$ for each of $i$’s trusted peers $j$ from one of the score managers, and then adds them: $R_{:,i} = (1 - \alpha) \sum_j S_{:,j} D_{j,i}$. We note here that the answering score manager computes $S_{:,j} = U V_{:,j}$. Alternatively, $i$ could request only a subset of rows of $S_{:,j}$ instead of the entire vector. For instance, when a query returns hits from a set of peers $L$, $i$ can request only the rows of $S_{:,j}$ corresponding to peers in $L$. 
3.3 Simulation Setup

We build our simulation model along the lines of EigenTrust. At a high level, the model represents a P2P file-sharing system consisting of honest as well as malicious peers downloading files from each other. Honest peers in such an adversarial network would want to download only authentic files. Conversely, malicious peers would want honest peers to download inauthentic files. To do so, malicious peers can resort to various attacks that potentially undermine the usefulness of the system, particularly to honest peers. To defend against such attacks, honest peers employ a trust-based algorithm to evaluate the trustworthiness of other peers. In the rest of the chapter, we use the percentage of inauthentic downloads as the basic metric to evaluate the performance of PET against the baselines ET and Non-Trusted (NT).

We now describe the simulation execution. An experiment proceeds in simulation cycles which in turn is divided into query cycles. During a query cycle, each honest peer $i$ in the network may either issue a query or be passive. When a query is issued, the peer is returned with hits from a subset of all peers in the network. The ratio of good and bad hits among them is equal to the proportion of honest and malicious peers in the network. The peer $i$ then selects one peer among these hits, based on latter’s trustworthiness, to download the file. Peer $i$ then rates the transaction as positive (+1) or negative (-1) based on whether the downloaded file was authentic or not. At the end of each simulation cycle, a trust-based algorithm is used to compute reputation scores which are later used in the subsequent simulation cycle for download source selection. Also, statistics such as the number of authentic and inauthentic downloads by each peer are collected. Each experiment is run 100 times over which the results are averaged.

We aim to evaluate how PET performs in diverse transaction models based on following distributions:

- **Random**: Content is randomly distributed among peers, and the queries answered by a peer are also random.
- **Community**: A peer usually answers queries issued by peers in the same community.
- **Power-law**: A few peers among the entire population have the most popular content, and answer majority of the queries issued in the network. The probability that a peer $i$ answers a query is proportional to $i^{-\gamma}$.

In each experiment in the next section, we simulate a network of 1000 peers comprising the same number of honest and malicious peers. Each query by an honest peer is returned with hits from 25 honest peers and 25 malicious peers (5% of the population). While the subset of honest peers answering a query is based on a particular transaction model, the subset of malicious peers answering the query is always based on power-law distribution.\footnote{We refer the reader to \cite{57} for more details.}
Table 3.1: Simulation Settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default/Range</th>
</tr>
</thead>
<tbody>
<tr>
<td># Total nodes</td>
<td>1000</td>
</tr>
<tr>
<td># Pre-trusted nodes</td>
<td>10</td>
</tr>
<tr>
<td>% Malicious peers</td>
<td>50%</td>
</tr>
<tr>
<td>Transaction model</td>
<td>random, community, power-law</td>
</tr>
<tr>
<td>% peers answering a query</td>
<td>5%</td>
</tr>
<tr>
<td>Communities</td>
<td>G1 (75%) and G2 (25%)</td>
</tr>
<tr>
<td>Power-law exponent $\gamma$</td>
<td>0.8 ([0.0,2.0])</td>
</tr>
<tr>
<td>Teleportation parameter $\alpha$</td>
<td>0.15</td>
</tr>
<tr>
<td># Simulation cycles</td>
<td>30</td>
</tr>
<tr>
<td># Query cycles</td>
<td>50</td>
</tr>
<tr>
<td># Runs for each experiment</td>
<td>5</td>
</tr>
</tbody>
</table>

where $\gamma = 1$. This enables malicious peers to gain a vantage point against honest peers. Table 3.1 shows various parameters and their default values or ranges used in the simulation.

### 3.4 Resilience to attacks

In this section, we evaluate the performance of PET under various attacks by considering that each peer trusts only itself. In other words, each peer $i$ computes its personalized reputation vector $\vec{r}_i$ by assigning $d_i[i] = 1$. In the next section, we discuss the role of social network-based trusted peers.

#### 3.4.1 Collusion with malicious spies

A subset of malicious peers acting as spies upload authentic files when selected as download source. At the same time, these spies assign high trust scores to the rest of malicious peers which always upload inauthentic files as well as form a malicious collective / ring among themselves.

Figure 3.2 plots the percentage of inauthentic downloads versus the percentage of spies among malicious peers. PET outperforms ET in various scenarios. We particularly note that ET performs worst in the community model. This is because peers in the community with no pre-trusted peers have low reputation scores, and hence differentiating them with low ranked malicious peers is difficult. However, both PET and ET perform very well in the power-law model, since only a few top ranked honest peers answer most of the queries.

#### 3.4.2 Eigenvector centrality attack

We now reformulate the eigenvector centrality-based attack, discussed in Sec. 3.1.3, to suit the current context of simulation environment. That is, malicious peers upload authentic files to topmost eigenvector central peers, while uploading inauthentic files to the rest of honest peers. At the same time, these malicious peers collude to form a ring among themselves.
Since the notion of centrality of a peer $j$ in PET is different from the perspective of each peer $i$ in the network, the strategy for malicious peers is non-trivial (unlike in ET). Malicious peers assign the values in the eigenvector of $(I - \alpha C^T)^{-1}$ to the corresponding peers, which are fundamentally their eigenvector centrality scores. The intuition behind this idea is that the globally topmost central nodes are also highly central from many peers’ individual perspective in the context of PET.

We now examine the impact of such an attack. Figure 3.3 plots the percentage of inauthentic files downloaded versus the percentage of topmost central nodes to which strategic malicious peers upload authentic files. We make the following observations. First, the impact of this attack is significantly stronger than of any of the previous attacks studied [57]. For instance, in the random model, the performance of PET under this attack is nearly three times worse than that under the above collusion threat. Second, PET outperforms ET in various scenarios. This result highlights the difficulty in attacking a personalized reputation system as compared to a global one. Third, for the random and community models, the performance of PET gradually decreases until malicious peers behave well with less than top 40% central nodes. However, the performance of PET improves when more central nodes are targeted but remains almost as bad as NT’s. Lastly, ET performs worse than NT in most cases. This is because many malicious peers gain a lot higher reputations than their honest counterparts, and hence they are often selected for download.

We use the centrality-based attack strategy for malicious peers in the experiments in the rest of the chapter.
3.4.3 Extended Traitor attack

In this attack, malicious peers ‘milk’ the reputation in that they alternate between (i) uploading authentic files for some time to gain reputation, and (ii) later uploading inauthentic files to harm the system. We simulate this attack by extending the centrality-based attack. For five simulation cycles, malicious peers upload authentic files only to the top-25% central nodes. For the next five simulation cycles, malicious peers upload inauthentic files whenever selected for download. This process is continued until the end of the experiment.

Figure 3.4 plots the percentage of inauthentic downloads versus the simulation round. PET outperforms ET in various scenarios. In the case of PET, the traitor attack is counterproductive for malicious peers. The percentage of inauthentic downloads significantly decreases as the time passes since the reputation scores of malicious peers diminish to the extent that they are less likely to be selected for downloading in subsequent simulation cycles. In the case of ET, the reputation scores of a few malicious peers which answer most queries remain so high that they lose little despite acting maliciously for a few rounds in random and community models. However, in the power-law model, since a few honest and malicious peers which answer most queries compete with each other, the reputation scores of malicious peers reduce well below those of honest peers when acting maliciously for a few rounds.

3.4.4 Impact on transaction models

Figure 3.5(a) plots the percentage of inauthentic downloads versus the percentage of the community with no pre-trusted peers (G2%). The performance of PET is the least affected by increasing G2%. This shows that PET can be very effective even for small, new, or
Simulation Round
% of inauthentic downloads
10
20
30
40
50
60
70
Random
5 10 15 20 25 30
Community
5 10 15 20 25 30
Power−law
5 10 15 20 25 30
NT
ET
PET

Figure 3.4: Impact of traitor attack.

growing communities. On the other hand, the performance of ET gradually worsens as G2% increases. This is because more peers are ‘further away’ from the pre-trusted nodes, and hence lesser reputed.

Figure 3.5(b) plots the percentage of inauthentic downloads versus the power-law exponent $\gamma$. Since $\gamma = 0$ is essentially equivalent to the random query model, we note that the performance of both PET and ET significantly improves as the transaction model moves from random to power-law distribution. This is because, in the power-law model, only a few peers answer a majority of the queries. This helps these peers gain higher reputation than malicious peers, thereby increasing the likelihood of getting selected for download.
3.4.5 Sensitivity to teleportation parameter $\alpha$

Figure 3.6 plots the percentage of inauthentic downloads versus the teleportation parameter $\alpha$. The larger the value of $\alpha$, the worse the performance of both PET and ET. This shows a long random walk from a trusted peer is more likely to end at a malicious peer than at an honest one. This finding is consistent with that of Whanau [65]. Hence, a short random walk is more suitable in such adversarial systems.

3.4.6 Impact on ranking of search results

We now focus on the presentation of search results to users which can potentially influence user experience. When a query returns hits from the network, it is vital to show the user the search results from honest peers ranked higher than those from malicious ones. To measure the accuracy of PET and ET at differentiating honest and malicious peers, we use the metric Area under Receiver Operating Characteristic (ROC) curve or $A'$ [45]. This metric represents the probability that an honest peer is ranked higher than a malicious one. $A' = 1$ indicates perfect ranking, while $A' = 0.5$ represents random ranking. We perform the same experiment as mentioned in Section 3.4.2 and present the results from a ranking perspective unlike previous experiments where the focus was on reducing inauthentic downloads in the system.

Figure 3.7 plots the area under ROC curve versus the percentage of topmost central nodes to which strategic malicious peers upload authentic files. The ranking for both PET and ET worsens as we target more top ranked nodes. When more than half the top ranked nodes are targeted, PET performs worse than NT which means that malicious peers are ranked higher than honest peers in the ranking of search results. However, ET’s ranking is
Figure 3.7: Quality of ranking of search results.

worse than NT’s in each scenario, implying that malicious peers are more likely to be chosen for download. This also explains the poor performance of ET in Sec. 3.4.2.

3.5 Role of Social Network

We now discuss how social networks fits in our PET design.

3.5.1 Bootstrapping into the network

Invitation

We propose that entry into the network is based solely on invitations. That is, a new user needs to get invited by an existing user (typically a friend or a trusted acquaintance) in order to join the network, similar to the initial phases in online social networks such as Orkut and Google+ as well as BitTorrent Darknets. This approach addresses two important challenges. First, when a new user installs a P2P client and runs it for the first time, the client usually contacts either a central server (e.g., Mainline DHT in BitTorrent and Azureus) or a set of fixed super peers (e.g., Tribler, Skype) in order to connect to existing peers in the network. If such a central entity is down or compromised, new clients will not be able to join the network. Using our approach, a new peer obtains contact details of the existing peers from its inviter. Second, the network gradually evolves around a core of honest peers who also have social relationships among themselves. In this way, our approach leverages the trust inherent in the social network among these peers.
Social network-based trusted peers

After joining the network, a new peer faces the cold start problem in that whom to trust in an adversarial system is a fundamental question. Personalized EigenTrust using Social Networks (PETS) enables each peer to choose its own trusted peers. For simplicity in our experiment, we consider a random transaction model where a new honest peer trusts one existing honest peer uniformly at random. As a result, the former computes the personalized reputation scores from the perspective of the latter. We note here that PET stands for the scenario where each peer trusts only itself, while PETS implies that a new honest peer trusts an existing (older) honest peer.

We simulate the bootstrap scenario with a network comprising 250 honest peers and 500 malicious peers for the first 10 simulation cycles, after which 250 new honest peers join the network. We measure the number of authentic and inauthentic downloads by new honest peers at the end of the experiment.

Figure 3.8(a) plots the comparison of the performance of each algorithm for new honest peers. PETS outperforms all other approaches including PET. In addition, PET performs nearly as worse as NT, while ET performs the worst. Hence, trusting an older honest peer during bootstrap phase helps in significantly reducing inauthentic downloads. As a result, it potentially improves user experience which is vital in retaining the users (especially the newcomers) in the network.

Choice of trusted peers

We now examine how the choice of trusted peers affects the performance for the new honest peers. We use a similar setup as the above experiment, except that the 250 new honest peers are divided equally into three categories based on their type of trusted peer. One-third of these newcomers trust only themselves and another one-third trust in an existing honest peer, while the trusted peers for the rest are all malicious.

Figure 3.8(b) plots the percentage of inauthentic downloads versus the type of trusted peer by a newcomer. We see that trusting an existing honest peer is better than trusting either a malicious peer or just itself. For new honest peers, this implies that keep your friends closer and your enemies farther.

3.5.2 Defending against Sybil attacks

Social networks have been leveraged by the research community to counter Sybil attacks [43] in open adversarial systems [40, 104, 105]. The fundamental assumption behind these social network-based sybil defense (SNSD) schemes is that, although a malicious entity may create large number of Sybil nodes, the number of social connections these Sybil nodes can form with honest nodes is very limited due to high social engineering cost. As
3.6 Related Work

Many previous studies have critically analyzed ET and proposed personalized versions of the algorithm [36, 37, 52]. Chirita et al. [36] argue that a subset of pre-trusted peers can
be chosen by peers based on their tastes, instead of a fixed set as in ET. They propose a distributed algorithm similar to Personalized PageRank [53] to compute personalized reputation scores for each peer. Its performance was shown to be similar to ET in their work, whereas our study shows that not only PET outperforms ET under diverse scenarios but also ET performs worse than NT when centrality-based attacks are applied. Jansen et al. [52] discuss the vulnerabilities of ET, particularly when the pre-trusted peers leave the system or are compromised. They propose Federated EigenTrust which employs elected ‘representative nodes’ in place of pre-trusted peers. While their concerns are valid, our study shows that, even in a churn-free and non-compromised system, ET fares poorly when the network exhibits community structure. More recently, Choi et al. [37] have presented Personalized EigenTrust which computes the normalized trust scores (refer to Sec. 3.1.1) using beta distribution, which is not our main focus in this chapter. We note that all these studies lack extensive analysis on how their algorithms perform under various transaction models and attack strategies.

Evaluation of ET on the network traffic dataset of the Maze P2P file-sharing system by Lian et al. [66] shows that ET produces less than ideal reputation scores for both honest and malicious peers under various collusion patterns. Based on the results of our study (refer to Sec. 3.4.5), we suspect that such poor performance can be explained by their use of a very high value of $\alpha$ (=0.9 in their work). The higher the $\alpha$, the higher the likelihood of a random walk escaping into the malicious region. It will be interesting to see the results of their experiments when run with lower values of $\alpha$. Here we note that $\alpha$ in our study is equivalent to $1 - \alpha$ in theirs.

### 3.7 Concluding Remarks

We have empirically shown that ET is highly vulnerable to community structure and eigenvector centrality attack, and also that its two main bottlenecks are pre-trusted peers and global ranking. To address these shortcomings, we propose Personalized EigenTrust which enables each peer to (i) calculate personalized reputation scores of other peers, and (ii) choose its own set of trusted peers from the social network among peers. The use of social network-based trusted peers simply eliminates the need for the pre-trusted ones, thereby making the system autonomous. In addition, the social network enables newcomers to bootstrap into the network seamlessly as well as defend against Sybil attacks. Our evaluation on synthetic networks operating under diverse transaction models (random, power-law, and community) and targeted attacks including the novel centrality attack reveals that PET outperforms ET in all these scenarios. The insights gained in our study have important implications to existing and future designers of flow-based reputation systems. For instance, pagerank- and maxflow-based trust schemes need to be evaluated under eigenvector- and betweenness-centrality attacks.
Table 3.2: P2P Transaction Networks: Analogies

<table>
<thead>
<tr>
<th>File-sharing</th>
<th>Marketplace</th>
<th>Service-oriented</th>
<th>Social media</th>
</tr>
</thead>
<tbody>
<tr>
<td>download file</td>
<td>transaction</td>
<td>subscription service</td>
<td>view video/photo/story</td>
</tr>
<tr>
<td>uploader</td>
<td>seller</td>
<td>provider</td>
<td>uploader</td>
</tr>
<tr>
<td>downloader</td>
<td>buyer</td>
<td>consumer</td>
<td>viewer</td>
</tr>
</tbody>
</table>

The concept of PET is very general and has wide range of applications in P2P transaction networks\(^3\) (Table 3.2) which typically attract malicious behavior, in both centralized and distributed systems. For instance, online content voting systems such as YouTube and Digg face attacks where poor content is rated highly by many adversaries, thereby attaining a space on Most Popular pages. To counter such attacks, PET can exploit both voting patterns and social network relationships of users to filter out bad content and uploaders, and recommend only relevant and trustworthy content to users.

---

\(^3\)Here, ‘P2P’ refers to relationships such as peer-to-peer, person-to-person and user-to-user.
Chapter 4

Leveraging Trust and Distrust for Sybil-Tolerant Voting in Online Social Media

Voting is a vital component of online social media (OSM). Votes on content items in OSM (e.g., likes in YouTube and Facebook, favorites in Flickr, and diggs in Digg) are typically incorporated into many of their central features such as recommendations, ‘most popular’-like pages and ranking search results. Besides popularity, voting helps in determining the trustworthiness of content. Furthermore, voting offers a much cheaper alternative to the actual content analysis and human oversight. This aspect is particularly significant taking into account the scale and rate at which content is published in current OSM.

Due to open membership access in OSM, voting in such systems is vulnerable to Sybil attacks [43]. Malicious users can create multiple cheap Sybil identities that outvote real users of the system. In this manner, Sybils can promote spam content even to the front page of an OSM by voting positively (e.g., Tube Automator for YouTube [9]), as well as demote trustworthy content by voting negatively on it.

SumUp [92] is the state-of-the-art approach to defend against such an attack. SumUp leverages the social network among users to limit the number of fake votes collected from Sybil identities to O(1) per attack edge, which is a trust edge between a malicious user and an honest user. Although this is substantially more robust than directly accumulating votes, we contend the resulting solution still leaves room for considerable damage by attackers. For instance, if malicious users constitute a small fraction (e.g., 1%) in a large network of 1 million users, SumUp would face more than 10,000 attack edges. Such a scale of the attack even with conservative estimates drives the main research question of the chapter: can the fake votes collected from Sybils in OSM be reduced to less than O(1) per attack edge?

In this chapter, we leverage both trust- and distrust relationships among users to limit the votes collected from Sybil identities. We model the system as a signed network, where posi-
tive edges represent trust relationships that are inherent in the social network among users in OSM, and negative edges represent distrust relationships between honest users, who identify some of the spam content items, and the Sybil identities that promoted them. The fundamental rationale of our approach is that the attack edges constrain the paths along positive edges between the endpoints of each negative edge. This is because any path between an honest node and a Sybil node along positive edges passes through at least one attack edge. Based on this rationale, we proceed in two phases. First, we build a resistance network to identify such attack edges and their endpoints. Second, we employ a vote limiter mechanism to limit the votes from Sybil identities whose paths to honest nodes along positive edges pass through the endpoints of attack edges.

Our evaluation based on simulations on large-scale networks of popular OSM shows both the feasibility of incorporating distrust alongside trust to defend against Sybil attacks, and that our method outperforms the current state-of-the-art approach, SumUp. Our results show that negative edges between a small fraction (10%) of honest users and Sybil identities are sufficient to identify endpoints of attack edges with a high probability. Moreover, our method is resilient to negative edges among honest users constituting one-third of all negative edges in the network. In addition, the fake votes collected from Sybil identities are reduced by a significant 80%, when compared to SumUp, by taking into account one negative edge each from just a quarter of all honest users.

The rest of the chapter is organized as follows. Section 4.1 discusses relevant previous studies. Section 4.2 presents the definitions, notations, and the datasets used in the rest of the chapter. Section 4.3 describes our approach to the problem at hand in detail. Section 4.4 discusses our evaluation, while the implications of the results and possible future extensions are described in Section 4.5.

4.1 Background and Related Work

Here we review two growing research topics, one each in the fields of distributed systems and social networks, which have not been studied together in detail.

4.1.1 Social Network-based Sybil Defenses

Open web-based and distributed systems are susceptible to Sybil attacks [43], where attackers can create multiple cheap Sybil identities in order to outvote and outcompete honest users. To defend against such attacks, a large body of work leverages social networks by incorporating their properties such as inherent trust relationships among users and graph structure into the designs of social network-based Sybil defenses (SNSD) schemes [40,65,84,91,92,104,105].
Each of these schemes makes two fundamental assumptions. First, although an attacker can create arbitrary number of identities, she cannot establish arbitrary number of trust relationships (attack edges) with honest users since forming a trust relationship requires high social engineering cost. This leads to a sparse cut between Sybil region containing malicious identities and non-Sybil/honest region containing honest users in the graph, which is then exploited by these schemes. Second, a social network graph is expander-like and fast-mixing \cite{76} in that a random walk in the graph quickly reaches a stationary distribution. Hence, a short random walk starting from a node in the non-Sybil region rarely escapes into the Sybil region.

These schemes can be broadly classified into two categories: Sybil detection and Sybil tolerance, as characterized in \cite{95}. Sybil detection schemes \cite{40, 65, 84, 91, 104, 105} depend solely on the graph structure to label nodes as Sybil or not. Although application-independent, these schemes run the risk of (i) false positives: an honest user is misclassified as a Sybil, thereby not granted any service, and (ii) false negatives: a Sybil is misclassified as a non-Sybil which may allow unwarranted and unlimited access. Sybil tolerance schemes \cite{74, 82, 92}, on the other hand, do not label nodes as Sybil or not. Instead, they limit the leverage of an attacker who may use multiple Sybil identities by exploiting both the graph structure and application-specific information.

In the context of voting in online content systems, SumUp \cite{92} is the state-of-the-art approach that leverages the social network among users to bound the number of fake votes cast by Sybils. Given a vote collector (VC), SumUp first defines a vote envelope by distributing limited number of credits along the edges in a breadth-first fashion starting from VC. Then, VC collects all the votes from voters within this envelope. Finally, VC collects the votes from voters outside this envelope who can find a distinct path (i.e., no two paths share a common edge indicating sufficient credit) to a node within this envelope. As a result, SumUp limits the number of fake votes collected from Sybils – attack capacity – to $O(1)$ per attack edge with a high probability. As argued earlier, although SumUp is substantially more robust than directly accumulating votes, the resulting solution still leaves room for considerable damage by attackers. We argue that the number of attack edges used in SumUp’s evaluation is conservatively small (e.g., 100 attack edges in a million-node network), particularly for very large networks of the scale of the real-world OSM.

Furthermore, recent studies highlight a couple of drawbacks of SumUp. First, the breadth-first credit distribution biases the vote envelope to include many high degree nodes, which consequently incentivizes them to create Sybil identities \cite{56}. Second, the performance of SumUp was observed to vary across many networks, highlighting the shortcomings of the envelope selection process \cite{97}. We note here that addressing these drawbacks is not the main focus in this chapter.

Although all the proposed SNSD schemes have considered positive trust relationships among users to defend against Sybil attacks, only a few however have explored the distrust
factor. MobId \cite{84} is a Sybil detection scheme that operates in a mobile network of devices. A device maintains a network of friends containing honest devices and a network of foes containing suspicious devices, using which it determines whether to accept or reject an unknown device. SumUp \cite{92} incorporates negative feedback from the VC to modify the credit distribution as well as penalize the links from VC to the malicious nodes. Nevertheless, neither study examines the distrust relationships among users other than those from the evaluating node. This implies that an honest node which has few distrust relationships to adversaries, a newcomer for instance, will be vulnerable to attacks from the latter.

### 4.1.2 Signed Network Analysis

Many online social networks comprise not only positive/trust relationships such as friendships among users but also negative/distrust relationships that indicate enmity or antagonistic feelings toward one another. Such networks are referred to as signed social networks (in short, signed networks). Most of these online social networks keep the distrust relationships such as ‘report as spam/abuse’, unlike friendships, private. However, a few such as Slashdot make these negative links in the form of ‘foes’ and ‘freaks’ publicly accessible to the outside world.

Recent studies on signed networks have predominantly focused on the edge sign link prediction problem: given a social network with signs on all edges, how accurately can we determine the sign of a hidden edge? In one study, Leskovec et al. \cite{63} employ social psychology theories of balance and status to predict the signs of edges. In another study \cite{62}, they develop a machine learning approach based on various combinations of triads for sign prediction. Kunegis et al. propose a number of spectral analysis techniques for sign prediction \cite{60}, and later extend them to clustering and visualization \cite{61} using Laplacian matrices of the signed graphs. More recently, DuBois et al. \cite{44} propose a promising technique for sign prediction which combines path-probability trust inference algorithm and spring embedding to infer network distance between nodes. Their rationale is that a node-pair with a positive edge attract each other, while a negative edge makes them repel.

Propagation of distrust, unlike trust, is a tricky issue. For instance, an enemy of an enemy is not necessarily a friend, while a friend of a friend can be considered trustworthy. Guha et al. \cite{48} propose that trust can propagate multiple steps whereas distrust propagates a single step. Kerchove et al. \cite{41} propose PageTrust, along the lines of PageRank, incorporating the knowledge of negative links while performing the random walk. In addition, various other studies explored propagating distrust in the network \cite{24, 72, 108}, none in the context of attacks.
4.2 Preliminaries

4.2.1 Notations and Definitions

Let $G = (V, E)$ denote an undirected and signed graph, where $V$ and $E \subseteq V \times V$ are the set of nodes and edges in the graph, respectively. An edge between nodes $i$ and $j$ is represented by $(i, j, s)$, where $s$ represents the sign of the edge. Subgraph $G^+ = (V^+, E^+)$ comprises only positive edges ($E^+ \subseteq E$), while subgraph $G^- = (V^-, E^-)$ comprises only negative edges ($E^- \subseteq E$); also, $V^+, V^- \subseteq V$.

Given a sample $S \subset V$ of nodes, the neighborhood $N(S)$ is a set of nodes connected to $S$ by positive edges, i.e., $N(S) = \{ w \in V - S : \exists v \in S \text{ s.t. } (v, w) \in E^+ \}$. Expansion of $S$ measures the number of neighbors of $S$ relative to its size: $\frac{|N(S)|}{|S|}$. Expansion quality of $S$ measures the extent to which the neighborhood of $S$ covers the network: $\frac{|N(S)|}{|V - S|}$.

4.2.2 Datasets and Mixing Times

Table 4.1 summarizes the datasets of popular OSM that we use in the rest of our chapter, to study the problem of voting in OSM under Sybil attack.

We examine the fast-mixing [76] property of these social network graphs, which is an assumption that many SNSD schemes [65,104,105] make, in order to explore the plausibility of incorporating this feature into the design of our method. The mixing times of four out of these six datasets were found to be ‘fast’ in [77]. For the sake of completeness, we examine the mixing times for all the six datasets using the same methodology as [77]. Specifically, given an initial distribution $\pi(i)$ at node $i$ and transition matrix $P^{(t)}$ which is the adjacency matrix with normalized rows, we compute the total variation distance $|\pi - \pi^{(t)} P^{(t)}|_1$ at each step (random walk length) $t$ where $\pi$ is the stationary distribution.

Figure 4.1 plots the total variation distance versus the walk length, averaged over 1000 initial distributions. The mixing times of all the graphs converge to stationary distribution in the order of $O(\log |V|)$, within a total variation distance of 0.2. This shows that the social network graphs of these OSMs have ‘fast’ mixing times. Put another way, these graphs exhibit expander-like properties [38]. In the next section, we discuss how the expansion property of the graphs is incorporated into our design.

4.3 Method

In this section, we first describe the problem we address along with a brief overview of our approach, and then detail the two main components of our method.
Table 4.1: Datasets of popular OSM. Notations: \(|V|\) and \(|E|\) represent the number of nodes and edges in the network respectively; \(\delta_{0.9}\) is the 90\(^{th}\) percentile diameter. The last column gives an example of a vote on an item in the corresponding OSM.

| OSM      | |V|  | |E|  | \(\delta_{0.9}/\delta_{1.0}\) | Vote/Item       |
|----------|----------|----------|----------|----------|---------------------------------|
| YouTube  | 1,134,890 | 2,987,624 | 7/15     | like/video       |
| Flickr   | 1,624,992 | 15,476,835| 7/14     | favorite/photo   |
| Digg     | 542,140   | 4,039,025 | 6/14     | digg/story      |
| Facebook | 3,097,165 | 23,667,394| 6/8      | like/page       |
| Epinions | 75,877    | 405,739   | 5/10     | review/product   |
| Slashdot | 82,168    | 543,381   | 5/9      | comment/story   |

Figure 4.1: Mixing times of social network graphs in OSMs. Random walks in Slashdot, Epinions and Digg converge to stationary distribution faster than in the rest, though all have short walk lengths.
4.3.1 System Model and Approach

System Model

We consider the problem of voting on content items in OSM under Sybil attack, inspired by SumUp [92]. The number of votes on a particular item in OSM is a salient factor considered in determining its popularity. Our model considers four types of users in the network. First, traitors are a fraction of users that turn opportunistic to game the system using Sybil identities to inject and promote spam content through fake votes. Honest (ie. non-traitors) users who have at least one trust relationship with a traitor are considered gullible, while honest users who have trust relationships only with other honest users are considered winners. Finally, a fraction of honest users who are vigilant identify some of the spam content items and vote negatively on it. As a result, a plausible interpretation is that these vigilant users and Sybil identities which have promoted spam content have fundamental disagreement on the quality of this content. Another perspective is that the relationship between these two sets of entities is that of mutual distrust.

We model such a system by an undirected signed graph $G$ (Fig. 4.2(a)), where nodes represent either users or identities and edges represent relationships between nodes. A positive edge represents a trust relationship, while a negative edge represents a distrust relationship. Attack edges are the trust relationships between gullible and traitor nodes, while defense edges are the distrust relationships between vigilant nodes and Sybil identities that have promoted spam content.

Adversaries can change the graph structure of the Sybil region besides add and delete the Sybil identities to suit their strategies. Since off-the-shelf algorithms in the area of signed networks predominantly depend on graph structure for their effectiveness, these algorithms cannot be directly applied to defend against Sybil attacks.

Goal and Assumptions

Our main goal in this chapter is to limit the number of votes collected from Sybils to less than the number of attack edges. We make the following assumptions that influence our design. First, $G^+$ is connected, i.e., any two nodes in $G$ are reachable along positive edges. Second, the number of traitor nodes is significantly less than the number of honest nodes. As a result, the number of attack edges is limited. Third, creating new attack edges is difficult due to high social engineering cost, similar to the assumption previous SNSD schemes [65, 104, 105] make. Lastly, all honest nodes have a consensus on spam content items, and thus a negative edge exists only between an honest- and a Sybil node. We relax this last assumption in Section 4.4.1.
Honest Region

Sybil Region

Winner

Gullible

Traitor

Vigilant

Defense Edge

Attack Edge

Sybil

Figure 4.2: [Best viewed in color.] A pictorial illustration of the system and approach. \[4.2(a)\] Honest region comprises winner (dark blue), vigilant (green) and gullible (light blue) nodes. Sybil region comprises traitor (orange) and Sybil (dark red) nodes. Defense edges between vigilant and Sybil nodes are represented by dotted red lines. \[4.2(b)\] Paths along positive edges (blue thick double-headed arrows) corresponding to each defense edge are constrained by gullible and traitor nodes, where red circles indicate high conflict.
Approach Overview

We make two observations based on the model and above assumptions. First, attack edges between gullible- and traitor nodes can be viewed as ‘bridges’ connecting honest- and Sybil regions, with gullible- and traitor nodes being ‘bridge nodes’ (Fig. 4.2(a)). Second, these bridge nodes and bridges constrain the paths along positive edges between two endpoints of a negative edge, with one endpoint each in the honest- and Sybil regions (Fig. 4.2(b)). Notice that any path along positive edges corresponding to a negative edge between an honest- and a Sybil node passes through a gullible- and a traitor node. With this intuition in mind, we

- use defense edges to build a resistance network that identifies bridge nodes and bridges (Sec. 4.3.2), and

- employ a vote limiter mechanism to subsequently limit the votes from Sybils whose paths to honest nodes pass across these bridge nodes and bridges (Sec. 4.3.3).

4.3.2 Resistance Network

We construct a resistance network $R$ by incorporating distrust relationships among the nodes. We model this network by a weighted graph of $G^+$ where the weight of each edge is initially set to 0. For each negative edge $(i, j, s < 0) \in E^-$ in $G$, we find a path between $i$ and $j$ along positive edges ($\in E^+). For each edge on this path, we increase the weight of the corresponding edge on the resistance network by 1. As a final result, the weight of an edge represents the number of paths corresponding to negative edges passing through it. This weight can be viewed as the resistance of the edge. The resistance of a node $v$, $R(v)$, then is the summation of resistance of each of its edges. From another perspective, the resistance of a node or an edge can be interpreted as the measure of its ‘conflict centrality’ in the graph. That is, nodes and edges with high resistance indicate the extent of conflict between nodes across them.

Such a resistance network has a number of desirable characteristics. First, it presents a global perspective of resistance of any given node/edge, which is particularly interesting for administrators/moderators. Second, it is fairly immune to dynamics in the network such as additions/deletions of nodes/edges. Third, in personalized voting systems, each vote collector can use the same ‘fundamental’ resistance network to adapt her vote aggregation mechanism to suit her specific needs. This allows for scalability, thus leveraging the resistance network does not hamper the system scalability.

We now describe the construction of the resistance network in a scalable fashion. It is important to note that finding a path between any two nodes of a negative edge along positive edges in a large network in an efficient manner is non-trivial. To address this challenge, we proceed in two phases. In the first phase, we build the backbone of the network, which is a small connected subgraph comprising nodes that have high connectivity properties to the
rest of the graph. In the second phase, we find a path between two endpoints of each negative edge in the graph using this backbone, which is then used to update the resistance network. Following are the details of these phases.

**Backbone Construction**

We employ the snowball technique to expansion sampling (XSN) [70] (similar to its variant [69]) which constructs a sample $S$ of size-$k$ such that its neighborhood $N(S)$ covers a large portion of the network even for a small sample ($|S| << |V|$). We refer to this sample as the backbone of the network. At its core, this technique works by adding a node $v$ from the neighborhood $N(S)$ to $S$ at each step such that $v$ contributes most new neighbors to the current sample $S$. Since the scalability of XSN was not discussed in [69,70] and non-trivial, we develop a dynamic programming method for XSN to allow it to scale for large networks.

Naturally, the efficiency of XSN depends on the network it is applied to. We thus examine how XSN fares on our datasets. We begin with a sample $S$ containing one random node in the graph, and then we iteratively add nodes to $S$ until its size $|S| = 5\% |V|$. We measure the expansion quality of $S$ after each addition of a node. Figure 4.3 plots the expansion quality versus the sample size relative to all nodes in the graph, averaged over 5 runs. All graphs exhibit high expansion, even for a small sample. For instance, a sample size of just $1\% |V|$ has expansion quality of 0.5 or more, indicating that $S$ is on average within 2 hops from any node ($\in V - S$) in the rest of the network. Also, increasing this sample size reduces the average hop distance between $S$ and $V - S$. Section 4.4.3 discusses the scalability of XSN.

Another benefit of using XSN is that the backbone is a ‘representative’ sample of the community structure in the network [70]. By adding nodes that best contribute to the expansion of the sample, XSN essentially includes nodes which act as ‘bridge nodes’ between communities. As a result, the sample consists of nodes from most (if not all) communities in
the network. In our context, this implies that the backbone (and surrounding nodes) is likely to contain many of the gullible and traitor nodes.

**Pathfinder Algorithm**

We exploit the expansion property of this backbone to find a path along positive edges corresponding to a given negative edge with endpoints \(i\) and \(j\). We begin two paths \(P_i\) and \(P_j\), one at each endpoint, and then add a node each to the paths at every step such that it increases the chances of meeting each other. We now detail this searching process. If the head of \(P_i\) (which is the last added element, say \(v_i\)) is not in the backbone \(S\), we add to \(P_i\) a neighbor of \(v_i\) in the large graph which either is in \(S\) or contributes to near maximal expansion of \(P_i\) chosen probabilistically. However, if \(v_i\) is on the backbone (\(v_i \in S\)), we add to \(P_i\) a neighbor of \(v_i\) from the backbone \(S\) which contributes near maximal expansion of \(P_i\). We note here that this reduces the search space, from the large graph to the backbone, by a great extent. Also, if either \(v_i\) or its neighbor is in \(P_j\) (\(v_i \in P_j\) or \(N(v_i) \in P_j\)), we merge \(P_i\) and \(P_j\) which gives us the path between \(i\) and \(j\). Similar process is simultaneously carried out on \(P_j\). Section 4.4.3 shows that this algorithm finds short paths efficiently in large expander-like graphs, where most nodes in the network are one or two hops from the backbone which in turn itself is small.

**4.3.3 Vote Limiter**

We now describe the vote limiter mechanism, built along the lines of SumUp [92]. Our main aim here is to limit votes from Sybil identities collected by honest nodes. Hence, for simplicity in this chapter, we assume a vote envelope comprising all honest nodes, and focus on limiting the votes from voters outside this envelope which are essentially traitor- and Sybil nodes. We define the following:

- **Attack capability** is equal to the ratio of the collected votes from nodes in the Sybil region and the number of attack edges between the honest- and Sybil regions.
- **Capacity** of a node \(v\) is equal to the maximum number of vote paths that can pass through the node, which is half its degree: \(C(v) = \lceil|N(v)|/2\rceil\).
- **Effective Capacity** of a node \(v\) is equal to the ‘downgraded capacity’ of the node taking into account its resistance \(R(v)\): \(EC(v) = \lceil C(v) * \alpha^{R(v)} \rceil\), where \(\alpha \in [0, 1]\) is a dampening factor which decreases the ability of a node to allow vote paths to pass through it based on its resistance. We use \(\alpha = 0.8\) if the node is not on the backbone, and \(\alpha = 0.95\) otherwise, since many paths along positive edges corresponding to negative edges pass through the backbone nodes more often than through the other nodes.
We now detail the vote collection process. First, each node is assigned a credit equal to its effective capacity, and each edge outside the vote envelope is allocated a credit of 1 unit. Next, a vote from a voter outside the vote envelope is collected if there exists a distinct path (i.e., no two paths of voters share a common edge) from the voter to a node within the envelope, with each node and edge on the path having credit greater than zero. When a vote is collected in this manner, the credit of each node and edge on this path is decreased by 1 unit. The final number of collected votes divided by the number of attack edges gives the attack capability of the adversaries.

4.4 Evaluation

We evaluate our method in this section, aiming to address the following questions:

- How well can we distinguish winner, gullible, and traitor nodes by incorporating negative edges (defense edges) between honest- and Sybil regions?
- How resilient is our method when there are negative edges among honest nodes in addition to defense edges?
- How robust is our method to limit the votes collected from Sybil identities to less than the attack edges?
- How scalable are the techniques of the backbone construction and the pathfinder algorithm?

We now describe the experimental setup used in the rest of the section. Given a dataset, let the size of the respective graph be $N$. We first create a partition of this graph into Sybil- and non-Sybil (honest) regions, along the lines of [65]. We select a small fraction of nodes ($3\% N$) as traitors uniformly at random; the remaining nodes constitute the honest region. For each attack edge to a gullible node in the honest region, a traitor creates a Sybil identity and forms a positive edge with it. We note here that creating more Sybil identities and/or forming edges among themselves is not going to improve their attack capability, since their vote paths to the vote envelope are constrained by attack edges. Next, we construct the backbone of the graph (of size $3\% N$) starting from a random node in the honest region. Subsequently, we build the resistance network by incorporating defense edges ($e_D = 20\% N$).

4.4.1 Resistance

We examine the resistance of three types of nodes – winner, gullible, and traitor – in the following two scenarios.
Impact of Defense Edges

In the first experiment, we vary the number of negative edges (defense edges) between nodes in the honest- and Sybil regions. For now, we do not consider negative edges within the honest region. We construct the resistance network incorporating these edges, and collect a random sample of 5000 of these nodes.

Figure 4.4 plots the resistance versus defense edges, averaged over 5 runs. We make the following observations. First, the resistance of gullible and traitor nodes is much higher than that of winner nodes. This enables one to differentiate winner nodes from gullible and traitor nodes with a high probability. In 3 out of the 6 datasets, it is possible to consistently distinguish the three types of nodes, and in 2 other it is possible to distinguish winner and traitor nodes. Second, the resistance of all nodes increase as the number of negative edges increase. Traitor nodes in particular, and gullible nodes to some extent, gain higher resistance when more honest nodes are vigilant. Third, the resistance of gullible nodes is less than that of traitor nodes, suggesting that the paths along positive edges corresponding to each negative edge are constrained by traitor nodes more than by gullible ones. This is because the number of traitor nodes is typically lesser than the number of gullible nodes.

Resilience to Negative Edges in Honest Region

We now add negative edges among honest nodes, in addition to negative edges between honest- and Sybil regions. In this experiment, we keep the defense edges fixed (20% N).

Figure 4.5 plots the resistance as we vary the number of negative edges among honest nodes. We draw the following conclusions. First, the resistance of traitor nodes still remains higher than that of winner and gullible nodes, even after including (10% N) half the number of defense edges. This shows the resilience of our method to negative edges among honest nodes. Second, as expected, the resistance of winner nodes is higher in this experiment than the previous one (see Fig. 4.4), albeit much lesser than that of gullible and traitor nodes. Third, the resistance of gullible nodes is comparable or higher than that of the traitor nodes only in a few cases (e.g., Flickr and Digg). This is not a major concern since our main aim to reduce to the flow across these high resistance nodes, be they traitors or gullible.

4.4.2 Attack Capability

We examine the robustness of the vote limiter mechanism by varying the number of defense edges. Figure 4.6 plots the attack capability versus defense edges. The attack capability is 1 when there are no defense edges, like the scenario of SumUp when the vote collector has no distrust relationships. Our method reduces the attack capability by nearly 80% taking into account defense edges, one each from only a quarter of all honest users, indicating the resilience of our mechanism to Sybil attack. Furthermore, the attack capability decreases
Figure 4.4: [Box-and-whisker plot] Resistance of winner, gullible, and traitor nodes as the negative edges between honest- and Sybil regions increase. Notice that the vertical axis is in log-scale, and its measure is Resistance+1, instead of Resistance. So as to avoid plotting logarithm of Resistance=0. Also, note that Resistance+1=1 for winner nodes in some cases is represented by a small vertical line. Indicating all winner nodes plotted for this sample have Resistance=0.
as defense edges increase, implying that the more vigilant honest nodes the fewer the fake votes collected from Sybil identities.

Let us now consider the scenario of Digg as an example, where we report values averaged over 5 runs. When the traitor nodes constitute $3\% N = 16,264$, the number of attack edges and thus the number of Sybil voters is $e_A = 221,073$. As a result, the number of fake votes collected is equal to $e_A$, when there are no defense edges ($e_D = 0\% N$). By using $e_D = 25\% N$ defense edges, our mechanism limits the number of fake votes collected by nodes in the honest region to $15\% * e_A = 34,405$, which is nearly double the number of traitors. When $e_D = 100\% N$, fake votes collected by honest nodes is $6\% * e_A = 15,029$, which is similar to the number of the traitor nodes. We note that the number of collected fake votes can be reduced further by a method that decreases the effective capacity more ‘aggressively’ than the one used in this chapter (refer to Sec. 4.3.3).

4.4.3 Scalability

We examine the scalability of the two building blocks of the resistance network: backbone and pathfinder algorithm. We perform our experiments on an HP commodity laptop with Intel Core i7 1.6GHz processor and 4GB RAM running Windows 7. The code is written using the SNAP library [8], and compiled as a single-threaded Visual C++ application.

The left-half of Table 4.2 shows the computation times of the backbone using Expander Sampling (XSN) technique as we increase the sample size, averaged over 5 runs. Our dynamic programming-based implementation of XSN is observed to scale to large networks. Even a sample size of 5% in a 1.6-million node network of Flickr takes less than 20 seconds. A 3-million node network of Facebook takes just over a minute to sample over 150,000 nodes for the backbone, whose expansion quality is close to 1 (Fig. 4.3) implying that every node is nearly just a hop away from the backbone. Although the backbone is not likely to change frequently, the system designers can choose to compute it periodically at a very low cost.

The right-half of Table 4.2 shows the average query times and path lengths for the pathfinder algorithm based on 1-million queries of random node-pairs. We make the following observations. First, the average query time in these large networks is less than 1-millisecond, even for a 3-million node Facebook network. Second, as expected, finding a path takes longer time as the graph size increases in general. However, the average query time on Facebook network is faster than that on Flickr even though the latter network is half the size of the former. The main reason is that the expansion quality of Facebook is much higher for the same sample size percentage than that of Flickr (Fig. 4.3), which allows better searching capability.
Table 4.2: Scalability of backbone construction (in seconds) and pathfinder algorithm (in $\mu$ seconds).

<table>
<thead>
<tr>
<th>OSM</th>
<th>1%</th>
<th>3%</th>
<th>5%</th>
<th>$\mu$s/Query</th>
<th>Avg Path Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epinions</td>
<td>0.262</td>
<td>0.324</td>
<td>0.377</td>
<td>89</td>
<td>6.773</td>
</tr>
<tr>
<td>Slashdot</td>
<td>0.418</td>
<td>0.496</td>
<td>0.561</td>
<td>104</td>
<td>7.460</td>
</tr>
<tr>
<td>Digg</td>
<td>3.160</td>
<td>3.887</td>
<td>5.257</td>
<td>320</td>
<td>6.656</td>
</tr>
<tr>
<td>YouTube</td>
<td>3.378</td>
<td>5.447</td>
<td>9.032</td>
<td>409</td>
<td>7.804</td>
</tr>
<tr>
<td>Flickr</td>
<td>10.127</td>
<td>13.440</td>
<td>18.202</td>
<td>734</td>
<td>7.413</td>
</tr>
<tr>
<td>Facebook</td>
<td>38.394</td>
<td>49.339</td>
<td>70.499</td>
<td>505</td>
<td>13.680</td>
</tr>
</tbody>
</table>

4.5 Discussion

Our study opens scope for improvements in existing systems (the top three points below) as well as research in multiple directions (the rest), some of which we discuss here.

**Vote envelope creation:** Creating the vote envelope around a given vote collector is not the main focus in this chapter. Based on our method, a natural extension for envelope creation is to consider both minimal expansion and the resistance network. Starting from the vote collector in the sample $S$ representing vote envelope, at each step, we add a node $v \in N(S)$ in $S$’s neighborhood to $S$ such that it minimizes (i) the expansion of the updated $S$, thus reducing the bias toward high-degree nodes, and (ii) the resistance of the updated $S$, thereby including nodes with low resistance.

**Decentralized setting:** While our method was discussed in a centralized setting in this chapter, we believe that it can also be extended to distributed systems. The relationship between a node and its blacklist containing other nodes that have acted maliciously in the past can be viewed as a defense edge in our context. One can incorporate such defense edges, for instance, in an existing social network-based Sybil-resilient DHT system such as Whanau [65], in order to further improve its robustness. In Whanau, as the number of attack edges increases, the number of messages required to find an authentic value for a given key also increases significantly. Failed lookups can be used as defense edges to further reduce (i) the ability of the adversaries to pollute the DHT routing tables, and (ii) the lookup times and network overhead to find an authentic key-value pair when there are many attack edges.

**Multiple-pair short path problem:** Our pathfinder algorithm, based on maximum expansion-based search [69, 70], shows promise in finding a path between any two nodes in a large network in an efficient manner. While the all-pair shortest-path problem is well studied, the usage of shortest paths in the real-world scenarios is not always desired. Nodes and edges on these shortest paths can get overloaded if paths of many simultaneous queries of node-pairs coincide on a few nodes and edges. Our algorithm offers an alternative particularly in the scenarios where the constraint of finding the shortest paths is less strict.
**Formal analysis**: Our study shows the feasibility of incorporating negative edges, not necessarily from the evaluating node, in the network to defend against Sybil attacks. A potential future work may explore the formal analysis of why our method performs considerably better than the state-of-art approach: SumUp. Extending this line of thought, can we build generic algorithms taking into account both trust and distrust against Sybil attacks?

**Privacy-preserving**: Most current SNSD schemes take into account all positive edges in the network for their robustness. Considering the promising results of our approach, a research question that naturally follows is: by incorporating defense edges, can a few positive links be hidden to preserve privacy and still attain a comparable performance to the a priori?

**Consequence of negative edges**: A downside of using negative edges is that it might antagonize the environment. Competing nodes for popularity, for instance, will use Sybil attacks and negative edges to reduce the ‘reputation’ of each other. A recent study [41] suggests the use of undirected negative edges instead of the directed ones so that it disincentivizes such ‘unethical’ behavior.

### 4.6 Conclusions

In the context of voting in online social media, this chapter leverages trust and distrust relationships to limit votes from Sybil identities. The fundamental rationale of our approach is that attack edges, gullible- and traitor nodes constrain the paths along positive edges between two endpoints of a defense edge. Based on this rationale, we first build a resistance network to identify such bottlenecks using negative edges, and then employ a vote limiter mechanism to limit the votes from Sybil identities whose paths to honest nodes pass through gullible and traitor nodes.

Our simulation results on large-scale OSM point to the feasibility and the scalability of this approach. Defense edges from a small fraction (10%) of vigilant nodes to Sybil identities are sufficient to differentiate the winner nodes from the gullible- and traitor nodes with a high probability. Our method is resilient to negative edges among honest users. The attack capability is around 0.2 in our method, compared to 1 in SumUp, which uses one defense edge each from a quarter of all honest users. Finally, the pathfinder algorithm on which our approach is based can find a path given any two nodes in large networks in less than a millisecond on a commodity laptop, showing that the resistance network can be built in a scalable fashion.
Figure 4.5: Resistance of winner, gullible, and traitor nodes as the negative edges among honest nodes increase.
Figure 4.6: [Best viewed in color.] Attack capability of Sybil identities as the defense edges increase.
Chapter 5

Distrust-based Sybil Detection

The ease of creating an account in an online social network (OSN) such as Facebook, Flickr, and YouTube has mixed results. On the upside, little effort to join the network has contributed to its massive growth. Any user can register within a matter of a few minutes, and form friendships, follow others, and upload content such as photos and videos at no cost. On the downside, an adversary may perform a Sybil attack [43], by creating multiple fake accounts cheaply with a malicious intent to disrupt the system.

Real-world OSNs have observed Sybil attacks in their networks. According to Facebook’s filings [16] in August 2012, 83 million illegitimate accounts were found in the social network out of its 955 million active accounts. Since Facebook’s major revenue comes from targeted advertising, such a large fraction of fake accounts can result in loss or reduced spending of advertisers, thereby harming their business. Regional OSNs such as Renren [102] and Tuenti [29] also experience such fake accounts. Link farming similar to web was found in Twitter [46] using fake accounts.

An attack of such magnitude necessitates OSNs to adopt a wide range of defense schemes. Challenge-response mechanisms such as Captchas typically limit the rate at which Sybil accounts are created. However, cheap crowd-sourcing techniques enable fast account creation. Machine-learning techniques such as SVM can classify whether an account is malicious or not based on its properties such as friendship request and accept frequencies and clustering coefficient. These techniques are effective in identifying Sybil accounts with behavior highly deviant from normal users. Nevertheless, a counter strategy by these accounts is to replicate the patterns of normal users which make it difficult for these techniques to differentiate honest users from malicious ones.

A large body of work leverages social networks by incorporating their properties such as inherent trust relationships among users and graph structure into the designs of social network-based Sybil defenses (SNSD) schemes [29, 40, 65, 84, 91, 92, 99, 104, 105]. Sybil-Rank [29] is the state-of-the-art Sybil detection scheme that was shown to outperform various other schemes in synthetic scenarios. It was also applied on a real-world OSN – Tuenti –
and helped identify many fake accounts. One of the limitations of SybilRank as well as most of the above schemes is its vulnerability to targeted attacks. To improve the effectiveness of these schemes, it is worthwhile to explore and exploit additional information in the system on top of the social network [103].

Inspired by the previous chapter, we adopt a similar approach to the more generic problem of Sybil detection in OSNs. We leverage trust and distrust relationships among users to first rank them on their trustworthiness and then differentiate honest users from Sybil identities based on the ranked list. We now provide an overview of our method. We model the system as a directed signed network, where positive and negative edges represent trust and distrust relationships, respectively. Similar to previous SNSD studies [29, 104, 105], we begin with an assumption that the trust relationships between honest users and Sybil identities (attack edges) are limited due to their requirement of high social engineering effort.

Our rationale is based on the one in the previous chapter: attack edges constrain the paths along positive edges between two endpoints of each negative edge. Based on this rationale, we proceed in two phases. First, we build a resistance network on top of the signed network such that each positive edge over which a path corresponding to a negative edge passes is annotated with the endpoints of the negative edge. Such annotations add accountability as to who initiated a negative edge. We also limit the number of such paths passing over a positive edge. This bounds the counter attack capability of the adversary who may want to initiate negative edges from Sybil nodes to honest nodes. Moreover, it addresses some of the limitations of the design of the resistance network in our previous chapter. Second, we adapt SybilRank to incorporate the resistance network to distribute trust in the network from a few seed nodes. Negative edges from honest nodes to Sybil nodes reduce the amount of trust to flow from the honest region to the Sybil region, enabling our method to differentiate honest and Sybil nodes with a high probability.

We perform our experiments on popular datasets used in previous SNSD studies [29, 77]. We first devise a targeted attack strategy based on eigenvector centrality [33] for the adversary. The state-of-the-art Sybil defenses perform worse than a naive defense under this attack, suggesting a big room for improvement in the robustness of such Sybil defenses. Under this attack, our method is seen to perform near-optimal when there is a growing negative feedback from honest users on Sybil identities: the more the distrust, the better the performance. Furthermore, our method is resilient to a counter attack strategy where Sybil nodes initiate negative edges toward honest users, provided honest users identify a few Sybils.

The rest of the chapter is organized as follows. Section 5.1 reviews various types of social network-based Sybil defenses. Section 5.2 briefly discusses the two inspirational studies that we use in our design. Section 5.3 defines the problem at hand, while Section 5.4 describes the method to address the challenges. Our evaluation is carried out in Section 5.5.
5.1 Related Work

Here we review the literature on SNSD schemes which can broadly be classified into following categories.

5.1.1 Sybil Detection Schemes

SybilGuard [105] is the seminal study in the field of Sybil defenses that leverage social networks and their properties (e.g., fast mixing) to counter Sybil attacks [43]. In SybilGuard, each node performs a random route of length \( w = \Theta(\sqrt{n \log n}) \). A verifier accepts a suspect if the former’s random route intersects with the latter’s. When the number of attack edges is \( g = O(\sqrt{n} / \log n) \), SybilGuard accepts at most \( O(\sqrt{n \log n}) \) Sybil nodes per attack edge with a high probability. SybilLimit [104] improves upon SybilGuard in that SybilLimit accepts at most \( O(\log n) \) Sybil nodes per attack edge with a high probability for \( g = O(n / \log n) \). In SybilLimit, each node performs \( r = O(\sqrt{m}) \) short random walks of length \( w = O(\log n) \), and gathers the knowledge of the final edge (tail) of each random walk. A verifier accepts a suspect if they both share a common tail.

Gatekeeper [91], with a strong assumption that the social network graph is a balanced expander-like network, accepts at most \( O(\log g) \) Sybil nodes per attack edge when \( g < O(n / \log n) \). In the worst case when \( g = O(n / \log n) \), both SybilLimit and Gatekeeper accept \( O(\log n) \) Sybil nodes per attack edge. Gatekeeper works by distributing \( O(n) \) tickets to the rest of the network in a breadth-first fashion from a few seed nodes. A verifier accepts those nodes that have received tickets above a threshold. We note that SybilGuard [105], SybilLimit [104], and Gatekeeper [91] were designed primarily for decentralized environment.

Schemes such as SybilInfer [40], SybilDefender [99], Latent community model [28] and SybilRank [29] leverage the knowledge of the entire social network in a centralized setting in order to identify Sybil nodes. SybilInfer [40] and Latent community model [28] use Bayesian inference techniques to differentiate honest nodes from Sybil ones. SybilDefender [99] proposes a scalable way to identify Sybil nodes as well as identify a Sybil community around a Sybil node. SybilRank [29] is the state-of-the-art approach which was shown to outperform many of the above Sybil detection schemes. We discuss more details on SybilRank in the Section 5.2.1.

A recent study [102] observed Sybils in Renren, a real-world OSN. The authors use an SVM classifier with features such as invitation frequency, incoming/outgoing friendship requests, and clustering coefficient to identify Sybils with a high probability. Such machine learning techniques are particularly useful to filter out identities with pattern behavior highly deviant from normal users. An obvious counter strategy for an adversary is to create multiple Sybil identities that mimic patterns of honest users and their network structure which keeps
them under the radar.

5.1.2 Sybil Tolerance Schemes

Sybil detection schemes [40, 65, 84, 91, 104, 105] solely depend on the graph structure to label nodes as Sybil or not. Although application-independent, these schemes run the risk of (i) false positives: an honest user is misclassified as a Sybil, thereby not granted any service, and (ii) false negatives: a Sybil is misclassified as a non-Sybil which may allow unwarranted and unlimited access.

Schemes such as Ostra [74], SumUp [92], and Bazaar [82] aim to limit the leverage of an attacker who may use multiple Sybil identities by exploiting both the graph structure and application-specific information. They employ maximum flow-based approaches to minimize the impact of Sybil nodes. In essence, these schemes limit the attack capability of an adversary based on its contribution to honest nodes. In this line of work, Viswanath et al. proposed Canal [96] that can perform maxflow-like computations in a scalable fashion.

Sybil tolerance schemes are highly suitable in scenarios such as online marketplaces and spam prevention in email systems where the cost of a bad transaction is heavy. However, applying such schemes in OSNs where a primary goal is to identify fake and malicious accounts and quarantine them is non-trivial.

5.1.3 Distrust-based Schemes

While all SNSD schemes use only trust relationships, only a few explored the distrust factor. MobId [84] and SumUp [92] incorporates negative feedback from the verifier alone to defend against Sybil attacks. Chapter 4 took a step further by leveraging distrust relationships between multiple honest users (not limited to the verifier) and Sybil nodes in order to limit the number of Sybil votes. Section 5.2.2 discusses more details. We note that Chapter 4 was a first investigation determining the feasibility of incorporating distrust for (application-specific) Sybil-tolerant voting. How the method can be applied to a more generic problem of defending against Sybil attacks was left as an open research question. We aim to address this challenge in this chapter.

5.2 Preliminaries

This section outlines two studies that inspire the problem definition and our system design. The first study is the state-of-the-art Sybil detection scheme: SybilRank [29]. The second study is our recent work [35] which leverages trust and distrust for limiting the attack capability of Sybil identities. In Section 5.4, we incorporate some of the lessons learned in the second study to improve the robustness of the Sybil detection scheme from the first study.
5.2.1 SybilRank

SybilRank [29] is the state-of-the-art Sybil detection scheme that has been shown to outperform previous popular Sybil defenses. SybilRank was also evaluated on a real-world online social network – Tuenti – and helped identifying a large number of fake accounts. The main rationale in SybilRank is that an early-terminated random walk [103] originating from an honest node has a higher degree-normalized landing probability to end at an honest node than at a Sybil node. Based on this rationale, SybilRank proceeds in the following three phases:

Propagating Trust

Given a few trusted seed nodes, SybilRank propagates trust from these nodes using power iteration. First, a total trust of $T_G > 0$ is assigned uniformly to $k > 0$ seed nodes, and zero trust to all other nodes. At each iteration, a node uniformly distributes its trust to its neighbors. Equally, this node gathers trust from its neighbors and the sum of the collected trust is the new trust for the node. Let $T^i(v)$ denote the trust value of a node $v$ after $i$ iterations.

$$T^i(v) = \sum_{(u,v) \in E} \frac{T^{(i-1)}(u)}{\deg(u)} \quad (5.1)$$

The power iteration is terminated after $O(\log n)$ steps, which is sufficient to reach a near-uniform distribution of degree-normalized trust over the honest region. At the same time, the amount of trust that escapes into the Sybil region is minimized due to limited attack edges. An analogous argument in SybilLimit [104] is that short random walks rarely escape into Sybil regions.

Ranking by Degree-normalized Trust

SybilRank computes degree-normalized trust for each node, after $w = O(\log n)$ iterations. The resultant score for a node $v$ is

$$\hat{T}_v = \frac{T^w(v)}{\deg(v)} \quad (5.2)$$

The authors argue that ranking the nodes by degree-normalized trust (i) reduces the bias toward high degree nodes, and (ii) limits Sybils ranked above honest nodes to $O(\log n)$ per attack edge.
Annotating the Ranked List

In the final phase, SybilRank annotates intervals in the ranked list which may have fake segments. Through manual inspection or additional information, these annotations enable OSNs to classify whether a node is Sybil or not.

5.2.2 Leveraging Trust and Distrust for Sybil-Tolerant Voting

Chapter 4 focused on the problem of voting on content items in online social media (OSM) under Sybil attack [43]. To address this challenge, both trust- and distrust relationships among users are leveraged to limit the votes collected from Sybil identities. We now briefly discuss the method and some of its limitations.

Approach Overview

The system is modeled as a signed network, where positive edges represent trust relationships that are inherent in the social network among users in OSM, and negative edges represent distrust relationships between honest users, who identify some of the spam content items, and the Sybil identities that promoted them. The fundamental rationale is that the attack edges constrain the paths along positive edges between the endpoints of each negative edge. This is because any path between an honest node and a Sybil node along positive edges passes through at least one attack edge. Based on this rationale, the method proceeds in two phases. First, a resistance network is built to identify such attack edges and their endpoints. Second, a vote limiter mechanism is used to limit the votes from Sybil identities whose paths to honest nodes along positive edges pass through the endpoints of attack edges. We now briefly describe the two phases.

Resistance Network

A resistance network $R$ is modeled as a weighted graph of $G^+$ where the weight of each edge is initially set to 0. For each negative edge $(i, j)$ in $G$, a path between $i$ and $j$ along positive edges is identified. For each edge on this path, the weight of the corresponding edge on $R$ is increased by 1. The resistance of an edge (a node) is then defined as the number of the number of paths corresponding to negative edges passing through it. Based on the above rationale, the expectation is that the attack edges and their endpoints would have higher resistance compared to other edges and nodes respectively.

Building the resistance network in a scalable fashion proceeds in two steps. In the first step, expansion sampling (XSN) [70] is used to construct a backbone of the network, which is a small connected subgraph comprising nodes that have high connectivity properties to the rest of the graph. In the second step, a path between two endpoints of each negative edge
in the graph is identified using this backbone, which is then used to update the resistance network.

**Vote Limiter**

The main aim of the vote limiter mechanism is to limit votes collected from Sybil identities. Each node is assigned a credit value such that, if its node degree is low or its resistance is high, then its credit is low. Each edge is assigned a credit of 1 unit. A vote from a voter is collected if there exists a distinct path from the voter to an honest node, with each node and edge on the path having credit greater than zero. When a vote is collected in this manner, the credit of each node and edge on this path is decreased by 1 unit. Since attack edges and their endpoints typically have high resistance, fewer Sybil votes are collected.

**Limitations**

We now discuss some of the limitations of the method. First, there are no bounds on the number of paths corresponding to negative edges that can pass through a node or an edge. This enables attackers to initiate as many negative edges as possible targeting honest users, thereby increasing the resistance of the honest region. Second, there is little accountability on who initiated a negative edge. Negative edges from nodes with poor reputation are valued the same as those from highly reputed nodes. Third, the method assumes the knowledge of all honest nodes. In realistic scenarios, the system designers may trust only a few honest nodes, and then infer whether any other given node is honest or Sybil.

### 5.3 Problem Statement

This section lays out our model, assumptions and goals for the rest of the chapter.

#### 5.3.1 System Model and Threat Model

We adopt system and attack models similar to those in previous SNSD studies [29,91,104,105]. The system consists of *n* honest nodes / identities, each representing a unique honest user. In addition, the system has one or more malicious attackers, each with one or more Sybil nodes / identities. All the Sybil nodes may collude with each other and simultaneously be controlled by an adversary.

There exists a directed signed social network (signed network, for short) $G$ among all nodes in the system. A trust relationship between any two nodes in the network is mutual, and thus denoted by two positive reciprocating (one in each direction) edges between them. The subgraph comprising all honest nodes is connected. The adversary may create arbitrary trust relationships among Sybil nodes. However, trust relationships (attack edges $g$) between
honest nodes and Sybil nodes are limited, since deceiving honest users to form friendships is non-trivial. Also, there exists a path along positive edges between any two nodes in the system. The adversary knows the entire social network.

If a node $P$ reports another node $Q$ as a fake account or for abuse, we consider such a relationship of distrust between the two nodes is not necessarily mutual. Hence, we denote such a relationship as a directed negative edge from the initiating node $P$ to the receiving node $Q$. Any node, be it honest or Sybil, can initiate a negative edge to another node in the system.

### 5.3.2 Assumptions

We make the following assumptions that influence our design. First, the social network graph is connected such that any two nodes can be reachable along positive edges. In other words, the subgraph $G^+$ comprising all nodes and their corresponding positive edges is a strongly connected component. Second, the number of attack edges is limited: $g = O(n/\log n)$. The adversary is able to trick only a few honest nodes. Third, creating new attack edges is difficult due to high social engineering cost, similar to the assumption previous SNSD schemes [65, 104, 105] make. Lastly, a negative edge exists only between an honest- and a Sybil node, since only Sybil nodes act maliciously. We relax this last assumption in our evaluation.

### 5.3.3 Goals

Our main goal in this paper is to improve the robustness of an existing Sybil detection scheme with minimal changes. The following are desired properties for our system design:

- **Effectiveness**: Sybil detection schemes aim to differentiate honest nodes from Sybil ones by essentially ranking each node [97]. Our system must rank honest nodes above Sybil nodes with a higher probability than the current schemes. At the same time, fewer honest nodes should be ranked below Sybil nodes.

- **Robustness**: The system must be robust to various attack strategies including targeted ones.

- **Complementary**: A current Sybil detection scheme must be able to incorporate our proposed design with little modification in order to improve its robustness.

### 5.4 System Design

This section describes our method to attain the above three-fold goals. We first discuss the approach of our method. Next, we describe how trust and distrust in the system can be
leveraged to build a generic framework using which an existing Sybil defense scheme can improve its robustness. Lastly, we show how such a framework can be incorporated into a state-of-the-art Sybil defense scheme: SybilRank.

5.4.1 Approach

Our rationale is based on the one in the previous chapter: attack edges constrain the paths along positive edges between two endpoints of each negative edge. This is because (i) any path between an honest node and a Sybil node along positive edges passes through at least one attack edge, and (ii) the number of attack edges is limited which is an assumption we make similar to previous SNSD schemes [29, 104, 105]. Based on this rationale, we proceed in two phases. First, we build a resistance network on top of the signed network such that each positive edge over which a path corresponding to a negative edge passes is annotated with the endpoints of the negative edge. Second, we adapt SybilRank to incorporate the resistance network while distributing trust in the network from a few seed nodes. Negative edges from honest nodes to Sybil nodes (defense edges) reduce the amount of trust to flow from the honest region to the Sybil region, resulting in low trust scores of Sybil nodes. This enables our method to differentiate honest and Sybil nodes with a high probability.

5.4.2 Resistance Network

A resistance network $R$ is built on top of the directed graph $G^+$ with all nodes and positive edges in the system. For each directed negative edge $(p, q)$, we find a path along positive edges from $p$ to $q$. Each directed positive edge $(u, v)$ on this path is then annotated with $(p, q)$. Over time, the positive edge $(u, v)$ on the resistance network would be annotated with a list of both endpoints of each directed negative edge whose corresponding path along positive edges passes through the edge $(u, v)$. Such annotations add accountability as to who initiated a negative edge. In contrast, our previous design of the resistance network (refer to Chapter 4) does not take into account the initiator of negative edges, and hence vulnerable to treating negative edges from highly reputed nodes as same as that of poorly reputed ones.

We limit the number of paths passing over a positive edge in the resistance network. This bounds the counter attack capability of the adversary who may want to initiate as many negative edges from Sybil nodes to honest nodes as possible. A consequence of applying bounds on links is that paths corresponding to new negative edges cannot pass through a positive edge whose quota is filled. To address this issue, we replace a negative edge from the annotated list only if its initiator is less reputed than that of the new negative edge. We note that the reputation of a node can be computed using any trust scheme.

For scalability, we compute the resistance network using the strategy from the previous chapter (also discussed in Sec. 5.2.2). Since paths along positive edges corresponding to a
negative edge pass more often on the backbone than the rest of the network, we choose a higher bound on the number of paths passing through backbone links than the rest.

5.4.3 Incorporating Distrust into SybilRank

We now describe how distrust can be incorporated to improve the robustness of SybilRank with minimal changes. Similar to SybilRank, we begin with a total trust of $T_G > 0$ uniformly distributed among $k$ seeds and zero trust among other nodes. At each iteration $i$, a node $u$ performs the following. First, the node computes its degree-normalized trust: $T^{(i-1)}(u)/\text{deg}(u)$. Second, the node $u$ counts the number of paths $p$ corresponding to negative edges passing through the directed edge $(u, v)$. We define $\beta(u, v) = p/2$ if the edge $(u, v)$ is in the backbone; otherwise $\beta(u, v) = p$. Third, the node $u$ distributes to its neighbor $v$ a trust of $T^{(i-1)}(u) \ast \alpha \beta(u,v)/\text{deg}(u)$, where $\alpha \in (0, 1)$ is a dampening factor. If the edge $(u, v)$ has no paths passing through it (i.e., $\beta(u, v) = 0$), the amount of trust $v$ receives from $u$ is $T^{(i-1)}(u)/\text{deg}(u)$, the same as in SybilRank. However, if one or more paths pass through the edge, $u$ distributes only a partial amount of the degree-normalized trust to $v$. Finally, the amount of trust that $u$ does not distribute to its neighbors is retained by $u$. This ensures that the trust in the system is conserved and thus the total amount of trust of all nodes remains $T_G$. The trust of a node $v$ after $i$ iterations is:

$$T^{(i)}(v) = \sum_{(u,v) \in E} \frac{T^{(i-1)}(u) \ast \alpha \beta(u,v)}{\text{deg}(u)} + T^{(i-1)}(v) - \sum_{(v,x) \in E} \frac{T^{(i-1)}(v) \ast \alpha \beta(v,x)}{\text{deg}(v)}$$  (5.3)

The first line in the above equation refers to the trust a node $v$ receives from its neighbors, and the second line refers to the trust retained by $v$ for the next iteration. Notice that if there are no negative edges in the network (i.e., $\beta(u, v) = 0$ for any node pair), the amount of trust $T^{(i)}(v)$ is the same as in the SybilRank scenario. The more the negative edges entering $v$ (high $\beta(u,v) > 0$), the lower $v$’s trust. The more the negative edges leaving $v$ (high $\beta(v,x) > 0$), the more trust $v$ retains.

5.5 Experiments

5.5.1 Setup

We employ an experimental setup similar to that in SybilRank.
Table 5.1: Network Properties. Key: $|V|$ = nodes, $|E|$ = edges, $\delta$ = effective diameter, $CC$ = clustering coefficient.

| Network       | $|V|$ | $|E|$   | $CC$ | $\delta$ |
|---------------|------|--------|------|----------|
| AstroPhysics  | 10,000 | 71,425 | 0.55 | 6.3      |
| Slashdot      | 10,000 | 122,933| 0.05 | 4.8      |
| Facebook      | 10,000 | 52,564 | 0.16 | 6.4      |
| Epinions      | 10,000 | 39,430 | 0.10 | 6.8      |

Datasets

We perform our experiments on some of the popular social network datasets used in SNSD studies [29, 77, 97]. Since some of these networks are directed in nature, we derive an undirected subgraph from the original graph such that each edge in the subgraph denotes two reciprocating edges in the original graph [78]. From the largest connected component of this subgraph, we in turn obtain a 10,000-node representative sample [71]. Table 5.1 summarizes network properties of these sample graphs.

Attack Model

For the Sybil region, we create five 1K-node regular random graphs with an average degree of 4. Attack edges connect honest and Sybil regions. The placement of these attack edges, in the sense that which honest nodes are tricked into forming trust relationships, depends on the strategy and the capability of an adversary and the gullibility of honest nodes.

Metrics

As argued by Viswanath et al. [97], SNSD schemes essentially assign trust scores to each node in the network and then rank them. To measure the accuracy of such a scheme at differentiating honest and Sybil nodes in the ranked list, we use the metric Area under Receiver Operating Characteristic (ROC) curve or $A'$ [45]. This metric represents the probability that an honest node is ranked higher than a Sybil one. $A' = 1$ indicates perfect ranking, while $A' = 0.5$ represents random ranking. To compute AUC, we randomly sample 500 honest nodes and 500 Sybil nodes in the network.

Experimental Settings

We assume the above 10K-node network as the honest region, and 10 random nodes in this region as seeds. We set the dampening factor $\alpha = 0.1$ since larger values of $\alpha$ result in more trust flowing into the Sybil region, as we show later in this section. The link bound is set
to 10, while the backbone link bound is set to 100. The results in the experiments in this section are averaged over 100 runs.

5.5.2 Centrality Attack

In this experiment, we examine how state-of-the-art trust schemes – SybilRank [29], Gatekeeper [91], and EigenTrust [57] – fare when a targeted attack is performed. SybilRank and Gatekeeper are top Sybil detection schemes in centralized and decentralized settings, respectively, while EigenTrust is the most well-known reputation algorithm that uses a power iteration method similar to SybilRank. We note here that, although SybilRank uses the degree-normalized trust scores (refer to Eq. 5.2) for final ranking, we nevertheless include the unnormalized trust scores (Eq. 5.1) for ranking as well.

We apply the centrality attack strategy inspired from Chapter 3 [33]. An adversary gathers the knowledge of the whole of the honest region as well as the seed nodes. Next, the adversary computes eigenvector centrality [23] personalized with respect to the seed nodes. The placement of attack edges is then targeted towards top-k ranked honest nodes at random. The number of attack edges is set to 1000 ($\approx O(n/\log n)$).

Figure 5.1 plots the performance of various schemes when the top-k ranked nodes are tricked into forming attack edges. The performance of each scheme degrades significantly with targeting higher ranked nodes. The performance is worse than random ranking (AUC=0.5) when top-1000 nodes are targeted for 3 out of 4 networks. Better performance for Slashdot can be attributed to its higher average degree. Among all schemes across most datasets, the unnormalized version of SybilRank performs better when top-1000 ranked nodes are targeted, while the normalized version of SybilRank performs worse than both Gatekeeper and EigenTrust. However, the former and the latter have similar performance when attack edges are targeted at random. This shows that the normalization step in the original SybilRank study [29] may not be necessary and sometimes even counter-productive against centrality attack.

In the rest of the section, we use a threat model of 1000 attack edges and centrality attack strategy on the top-500 ranked nodes. Notice that under this attack scenario, the performance of all schemes is worse than random ranking (AUC < 0.5) in AstroPhysics, Epinions, and Facebook and marginally better in Slashdot (AUC < 0.75). This presents a big room for improvement in the robustness of a defense scheme.

5.5.3 Defense Edges

In this experiment, we add negative edges from honest nodes to Sybil nodes. We assume paths along positive edges corresponding to these negative edges pass uniformly across all attack edges. We use the unnormalized version of SybilRank to compute reputations of
nodes for the replacement scenario when the annotated list on an edge reaches its bound and a new negative edge path is considered.

Figure 5.2 plots the performance of our method versus the number of negative edges from honest nodes to Sybil nodes (defense edges). The performance drastically improves with only 5000 defense edges, which is essentially equivalent to one out of every two honest nodes identifying a Sybil node. With more defense edges, the performance is near-optimal with AUC close to 1. However, the additional gains are modest.

### 5.5.4 Adding Sybil-to-Honest Negative Edges

In this experiment, we add negative edges from Sybil nodes to honest nodes alongside defense edges. Figure 5.3 plots the performance versus the number of negative edges from Sybil nodes to honest nodes, with various defense edges’ scenarios. The performance decreases with increase in negative edges from Sybils; yet, the performance is much higher than the unnormalized SybilRank. With more defense edges, the performance only degrades marginally even with large number of negative edges from Sybils. For instance, with as few as 2500 defense edges against 150,000 counter attack edges, our method performs much better than the unnormalized version of SybilRank. This indicates that, despite negative edges from Sybils to honest nodes, the defense edges originating from honest nodes is vital for high performance.
Figure 5.2: Varying the defense edges.

Figure 5.3: Varying counter attack edges.
5.5.5 Sensitivity to $\alpha$

We now focus on the parameter sensitivity to dampening factor $\alpha$. Figure 5.4 plots the performance versus $\alpha$, when the number of defense edges is 5000 and negative edges from Sybil nodes to honest ones is 10000. The larger the value of $\alpha$, the worse the performance. This is because higher $\alpha$ allows more trust to flow into Sybil region when compared to lower $\alpha$, even with same number of defense edges. Hence, lower $\alpha$ values are preferable for our scenarios.

5.5.6 Varying Link Bounds

Figure 5.5 plots the performance versus the link bound, while varying the backbone link bound. With increases in backbone link bounds greater than 50 and link bounds greater than 5, the performance either remains same or increases marginally. For low values of backbone link bound, the performance is both poor and less predictable. Hence, our settings of backbone link bound of 100 and link bound of 10 for the above experiments proved to be a reasonable choice.

5.6 Conclusion

This chapter tackles the problem of malicious Sybil accounts in OSNs. We describe how trust and distrust in the social network among users can be leveraged to build a generic
framework, using which an existing social network-based Sybil defense scheme can improve its robustness. We show how such a framework can be incorporated into a state-of-the-art Sybil detection scheme – SybilRank [29] – with minimal changes. Our evaluation on OSN datasets show that our method significantly outperforms SybilRank even under targeted attacks.
Chapter 6

Social Networks Meet Distributed Systems

There is a growing notion in the research community that online social networks (OSNs) and distributed systems complement each other in their respective challenges. OSNs such as Facebook and Twitter, despite their widespread adoption, are increasingly facing privacy concerns among many users. To address these concerns, multiple research initiatives [26, 39, 51] propose decentralized architectures for OSNs where users have complete ownership/rights of their own data, and their activity is not tracked easily. Distributed systems such as Chord and BitTorrent, on the other hand, are vulnerable to Byzantine failures and Sybil attacks [43]. Recent studies propose leveraging the trust inherent among users in OSNs to improve the robustness of such systems. A plausible argument – creating new trust relationships is expensive due to high social engineering costs – inspired the designs of generic decentralized Sybil defenses such as SybilGuard [105], SybilLimit [104], and Gatekeeper [91], as well as Sybil-resilient systems such as Whanau [65]. Another trend is the use of social networks to build privacy-preserving applications: secure lookup services such as X-Vine [75] and private data sharing systems such as OneSwarm [50].

At the heart of such systems, there exists a distributed overlay network constructed out of trust relationships of the social network among users. Each user is typically “bootstrapped” into the system from an OSN (e.g. Facebook). A common, though not explicitly stated, assumption in these studies is that the social network graph in the distributed environment would resemble its centralized counterpart. Further, we note that all major studies in this field used OSN datasets. We argue that not all users in an OSN might adopt the distributed system due to various reasons such as privacy concerns or lack of interest. The resultant distributed social graph could have different characteristics compared to its centralized version. As a consequence, the assumptions fundamental to these systems are potentially weakened.

Designing a practical social network-based distributed system requires the understanding of the characteristics of an operational distributed social network. With such a design...
goal in mind, we study various properties of Yahoo! Instant Messenger (YIM) which is a distributed social network in the wild used by millions of users. We make the following observations from the network. First, the YIM social network graph exhibits fast mixing property, which makes it suitable for Sybil defenses such as SybilLimit [104] and Whanau [65]. Second, only 5% to 15% of all users are online at any instance in a day, with most users having very short sessions, indicating the heavy churn in the network. Third, there is a strong correlation between the number of friends a user has, her online time, and activity in the network.

We examine the implications of these properties on social network-based distributed systems particularly from an overlay perspective. More specifically, we look at the impact of heavy churn on the connectivity of the trust overlay network. Our analysis on the Yahoo social network graph reveals that the overlay disintegrates into multiple disconnected components under a churn of 85% to 95% as observed in the Yahoo network. Moreover, a node remains disconnected nearly all the time under such a high and realistic churn. This insight motivates the need for a new design that improves the connectivity of the network to handle heavy churn.

We explore a straw-man solution to address this problem of connectivity. We find that leveraging all trust relationships from the two-hop neighborhood nodes – friends of friends – of each user significantly improves the connectivity compared to its one-hop counterpart even under heavy churn. However, our analysis also shows that the consequence of exploring the two-hop neighborhood in order to improve the connectivity makes the system vulnerable to an attack of an order of magnitude greater than a simple one-hop solution.

In this chapter, we explore a new design point in this tradeoff between network connectivity and attack resilience. Our main goal in this chapter is to improve the connectivity of the network under heavy churn while incurring a marginal security cost. The fundamental rationale of our approach is two-fold: (i) connecting to nodes contributing to greater expansion improves the network connectivity, and (ii) connecting to nodes contributing to lower conductance improves the attack resilience. We propose an adaptive 2-hop method that only adds a few specifically selected two-hop neighbors that contribute to higher expansion and lower conductance. Adding only a few extra edges on top of the (direct) social network reduces the attack capability of an adversary.

We evaluate our method based on the traces of the YIM network under a variety of trying conditions. Our investigation of the tradeoff between network connectivity and attack resilience reveals that our method significantly improves the network connectivity even under heavy churn by the addition of a very few select nodes (less than 5) at the cost of a modest increase in the attack capability of the adversary. Our method is also seen to be resilient to heavy churn with a performance comparable to the above straw-man solution under various scenarios: (i) steady state where the number of nodes that join the network is equal to the number of nodes that leave, (ii) flashcrowd where joining nodes exceed leaving
nodes resulting in a rapidly growing network, and (iii) recession where more nodes leave the system than the joining ones.

In the wide spectrum of the straw-man solution providing strong network connectivity under heavy churn and the one-hop solution providing strong attack resilience, our adaptive 2-hop method fills an important point in this design space making it significantly more robust to heavy churn than one-hop solution and with a modest attack resilience.

We now describe the outline of the chapter. Section 6.1 presents the details of an operational distributed social network. Section 6.2 examines the implications of the properties of a distributed social network, from an overlay perspective. Section 6.3 explores a straw-man solution to improving the network connectivity and its consequences. We propose our method in Section 6.4 to improve network connectivity at a low cost. Section 6.5 describes the evaluation of our method. Section 6.6 review the literature in this field.

6.1 A Distributed Social Network

Yahoo! Instant Messenger (YIM) is an instant messaging client used by millions of users everyday. A typical user first installs the YIM client on her desktop, and then adds friends and acquaintances who are also using YIM as ‘contacts’ in order to chat with them. This whole ecosystem of YIM users in a decentralized setting and their friendship relationships can essentially be viewed as an operational distributed social network. This section examines various characteristics of this network, particularly with an aim to help build practical trust overlays.

6.1.1 Datasets

We obtained three datasets through the Yahoo! Webscope program which describe: (i) the social network relations among users [11], (ii) how users communicate with each other over a month [13], and (iii) how active they are during a day [12]. These datasets are completely independent of each other in the sense that users across datasets are not common. Since all the datasets belong to the same network, we believe that we can draw reasonable inferences of the system as a whole.

Social Network

The Yahoo! Instant Messenger friends connectivity graph dataset [11] contains a sample of the YIM ‘friends graph’ from 2003. A node in this graph represents a YIM user, while an edge denotes the contact between two users. This contact can be interpreted as a friendship or a trust relationship between the two users. This undirected graph contains 1,878,736 nodes and 4,079,161 edges, and it is connected. In this chapter, for simplicity, we study the
properties of a representative subgraph (100K nodes) of the original graph which we derived using the expansion sampling (XS) technique [71].

**Interaction Network**

The Yahoo! Messenger user communication pattern dataset [13] captures the knowledge of who-messaged-whom among 100,000 users from 5649 zip codes every day from April 1 to 28, 2008. In each day, only the first communication between two users from a unique locale is logged: \((u_1, z_1, u_2, z_2, t)\), where \(u_i\) is a user in the zip code \(z_i\) at time \(t\). If the sender sends another IM in the day to the same receiver from the same sender zip code, the event is not logged. However, if either the receiver’s or the sender’s zip code changes, the event of the first subsequent IM is recorded.

We model the communications between users as an undirected unweighted graph which we refer to as interaction network [94]. A node in this graph represents a YIM user, while an edge denotes one or more messages sent between two users. In this chapter, we study the properties of the largest connected component of the interaction network which contains 81,219 nodes and 265,359 edges.

**Protocol Events**

The Yahoo! Messenger Client-to-Server protocol events dataset [12] contains events logged by 4,665,328 unique users in a 24-hour period in June 2010. This dataset also contains message events on who-messaged-whom at what time, but only during a day. Around 200 protocol events such as, ‘logging in’, ‘sending IM’, ‘accepting buddy invite’, ‘sending files’ are recorded with the associated timestamp and anonymous userid(s). However, the type of an event is not recorded.

**6.1.2 Graph Structure**

We examine the graph structure of the social- and interaction networks of Yahoo in comparison with those of Facebook, de facto standard OSN used in previous works [75,77], to better understand the topological differences between a distributed social network and an OSN. The main property we are interested here is the mixing time [76] of the network: a property upon which current works on robust and secure social network-based distributed systems [65,75,104,105] rely for their effectiveness. We also note that fast mixing time of a graph indicates its expander-like properties [38]. To the best of our knowledge, ours is the first work in this field to study the mixing property of an operational distributed social network.

We compute the mixing times for all the networks, whose properties are summarized in Table 6.1 using the same methodology as [77]. Specifically, given an initial distribution \(\pi^{(i)}\)
at node $i$ and transition matrix $P^{(0)}$ which is the adjacency matrix with normalized rows, we compute the total variation distance $|\pi - \pi^{(i)} P^{(t)}|_1$ at each step (random walk length) $t$ where $\pi$ is the stationary distribution. The speed with which the total variation distance decays is an indicator of the fast mixing property of the network.

Figure 6.1 plots the total variation distance versus the walk length, averaged over 1000 initial distributions. The mixing times of social network graphs of Yahoo and Facebook converge to stationary distribution in the order of $O(\log |V|)$, within a total variation distance of 0.2 indicating ‘fast’ mixing times. However, interaction networks of Facebook and particularly Yahoo are relatively ‘slow’ mixing. This result suggests that interaction networks exhibit stronger community structure in comparison to the corresponding social networks, which is consistent with a previous study on a Facebook network [100].
6.1.3 System Dynamics

Our dataset does not contain explicit information about the begin and end times of user sessions. To circumvent this limitation, we infer sessions based on user inactivity. As it is known that the keep alive messages have a periodicity of one hour in YIM [54], we consider that if a user sends at least one message in the n-th hour of the trace but not in the (n+1)-th hour, this user has a session ending in the (n+1)-th hour. The possible side effect of this heuristic is that in some situations where a user goes offline and comes back online in less than an hour, the two sessions will be considered as a contiguous one. Nevertheless, our analyses in this work are not concerned with the number of sessions a user has, or with very-short-term dynamics. Therefore, the expected effect of considering these two sessions as one is very limited. With this assumption in mind, we next study the dynamics of YIM clients.

Figure 6.2(a) shows the average percentage of users who are online during a particular hour with respect to all the users who are online at any instant during the 24-hour period. The analysis of the 24-hour period can be divided into 4 parts. First, from 4 AM to 9 AM, we see a flashcrowd of users joining the network with 4% of online users at 4 AM to 15% at 9 AM. Second, from 9 AM to 4 PM, nearly 15% of users are online every hour with a marginal drop at 2 PM which may be due to users having lunch during this period. Third, from 4 PM to 9 PM, online users reduce to 13% at 6-7 PM and increase back to 15% at 9 PM. This drop may be due to users leaving for their homes and having dinner, while the rise is perhaps post-dinner online time. Fourth, from 9 PM to 4 AM, online users drop from 15% at 9 PM to 4% at 4 AM.

Figure 6.2(b) plots the average percentage of users online in the previous hour continue to be online in the current hour. From 10 AM to midnight, nearly 40% of users are observed in subsequent hourly time intervals. This shows that users are often online for more than an hour during daytime. Similar to the pattern of online users each hour, late night sees a sharp drop in users being online across each hour with nearly 20% of them being common from 5 AM at 6 AM. In the morning until 10 AM, online users transitioning from one hour to the next increases to about 40%.

Figure 6.2(c) plots the percentage of users observed in k hourly time intervals. About half of all users are seen only in one time interval and nearly 90% of all users are observed in 5 or fewer time intervals, while less than 5% of all users are seen in 10 or more time intervals. This shows that user sessions typically last less than an hour. Such session times are comparable to those in other P2P systems [90].

Figure 6.3(a) plots the number of messages sent by a user versus her online time. As expected, the longer a user is online, the more messages she sends. However, users who are online for more than 20 hours send fewer messages. We suspect that such users may keep their clients running nearly all day while only occasionally chatting with others.
Figure 6.2: Hourly dynamics and user sessions.
We examine the relationship between user activity and friendships over a period of one month. Figure 6.3(b) plots the number of messages sent by a user versus the number of her friends. The more the friends, the more the messages a user sends. Figure 6.3(c) plots the number of unique days a user has been online in that month versus the number of her friends. The more the friends, the more time a user is online. Users with friends greater than 100 are few, which explains the scattered points on the rightmost part of the figure.

6.1.4 Summary

We now summarize our findings and discuss their implications:

- Yahoo’s social network graph, unlike its interaction network, exhibits fast mixing property, which makes it suitable for Sybil defenses such as SybilGuard [105], Sybil-Limit [104] and Whanau [65] in a decentralized environment.

- Percentage of all users online during a 24-hour period hits a peak of 15% during the daytime and a bottom of nearly 5% during late night. This suggests that (i) there are only a small fraction of all users are online at any instance, and (ii) the online pattern follows a non-stationary distribution.

- Almost 50% (90%) of all users are observed for less than 1 hour (5 hours) in a day. This indicates that most users are online for a short period.

- Nearly 20%-45% of users observed in one hour are also online in the next hour. This means that, although the online activity is dynamic as a whole, there exists a small fraction of users contiguous across one hour to the next.

- The more friends a user has, the more time she spends online and the more messages she sends. This shows that there is a strong correlation between the number of friends a user has, her online time and activity.

6.2 Effect of Churn on a Trust Overlay

Connectivity is a fundamental requisite for the proper functioning of social network-based distributed systems such as Whanau [65], X-Vine [75], and OneSwarm [50]. In such systems, users are connected by a trust overlay network which is constructed out of the social network relationships among users. The underlying rationale is that a link between two nodes in a trust overlay network is more secure and robust than the one between two mutually unknown nodes in systems such as Kademlia and BitTorrent. Based on this rationale, all their operations such as lookups and data transfer are performed on top of the overlay network, ensuring system performance.
Figure 6.3: Relationship between user activity, friendships, and online time.
Table 6.2: Distributed systems built on OSNs.

<table>
<thead>
<tr>
<th>Topic</th>
<th>System</th>
<th>Dataset(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sybil defense</td>
<td>SybilLimit [104]</td>
<td>Friendster, LiveJournal, DBLP</td>
</tr>
<tr>
<td></td>
<td>Gatekeeper [91]</td>
<td>YouTube, Digg</td>
</tr>
<tr>
<td></td>
<td>Whanau [65]</td>
<td>Flickr, LiveJournal, YouTube, DBLP</td>
</tr>
<tr>
<td>Privacy</td>
<td>X-Vine [75]</td>
<td>Facebook</td>
</tr>
<tr>
<td></td>
<td>OneSwarm [50]</td>
<td>last.fm</td>
</tr>
</tbody>
</table>

The performance of these systems, however, has not been evaluated under realistic churn rates. Studies such as SybilGuard [105], SybilLimit [104], and OneSwarm [50] have little explored how churn affects their performance. Other systems such as Whanau [65] and X-Vine [75] were put to test with a worst-case churn rate of 20%, i.e., 80% of all nodes are online at that moment. In contrast, our previous section shows that only 5%-15% of all nodes are online at any instance, which is equivalent to a churn rate of 85% to 95%, in an operational system. Moreover, all majors studies in this field used OSN datasets for their evaluation, as summarized in Table 6.2. In addition, the average degree in OSN datasets used in these studies is typically much higher than that of a distributed social network (Yahoo) which we observed in the previous section.

A natural question follows: what is the impact of high churn rates and low average degree in a distributed social network on the connectivity of its trust overlay network? We now examine their impact on the connectivity of the social network graph of Yahoo, in comparison with that of Facebook. We begin with a fully connected graph comprising all \( n \) nodes. Next, for a churn rate \( c \), we select \((1 - c) \times n\) nodes as being online each with a probability proportional to the power \( \gamma \) of its node degree. Notice that \( \gamma = 0 \) implies that each node has an equal probability of being online regardless of its degree. Larger \( \gamma \) values indicate that larger degree nodes tend to remain online longer than lower degree nodes. Finally, the rest \( c \times n \) nodes are considered offline and hence removed from the graph. In order to measure the connectivity of the online nodes, we define the disconnection probability as the difference of 1 and the ratio of the size of largest connected component of the graph and the graph size. The fewer the nodes in the largest connected component with relative to all online nodes, the greater the disconnection probability.

Figure 6.4 plots the disconnection probability versus the churn rate for Yahoo and Facebook social network datasets, with varying \( \gamma \). Disconnection probability grows with churn rate, indicating greater proportion of online nodes being disconnected from the largest connected component when more nodes are offline. Disconnection probability, however, grows at a much lesser rate with higher \( \gamma \) values. When larger degree nodes are online more often than the low degree nodes, the graph is connected to a greater extent. For instance, at churn...
rate $c = 0.9$, the probability that a random online node in the Yahoo network is disconnected from the largest connected component is around 0.5 when $\gamma = 1$. Almost every node in the Yahoo network is disconnected at this churn rate when $\gamma = 0$. In contrast, the resilience to disconnection in Facebook is much higher than in Yahoo. At a churn rate of $c = 0.95$ and $\gamma = 1$, the disconnection probability of a Facebook node is about 0.1 compared to 0.7 in Yahoo. Better resilience can be primarily attributed to higher average degree of Facebook compared to that of Yahoo.

In essence, our result highlights the disintegration of the trust overlay network into multiple disconnected components under high and realistic churn rates.

### 6.3 Improving Connectivity

Previous section motivated the problem of connectivity in a trust overlay network under high churn rates. In this section, we first explore a straw-man solution to improve the connectivity of the network under heavy churn, and then examine the consequences of such an approach.
6.3.1 Two-hop Neighborhood

We noted from Figure 6.4 in the previous section that Facebook is more resilient in terms of connectivity under high churn rates when compared to Yahoo, primarily due to the former’s much higher average degree than the latter’s. A simplistic approach to improving the connectivity of a low average degree network such as Yahoo is to explore nodes that are beyond the immediate trust circle of each node. We begin with tapping into the potential of the two-hop neighborhood of a node: the neighbors of its immediate neighbors. The idea is that trust is usually transitive, i.e., friends of friends are often considered trustworthy albeit to a lesser extent. Forming trust relationships with such friends of friends not only increases the degrees of the nodes but also potentially the connectivity of the network.

We examine how adding links to nodes in the two-hop neighborhood of each node affects the network connectivity under heavy churn. Figure 6.5(a) plots the disconnection probability versus the churn rate comparing both 1-hop and 2-hop scenarios with $\gamma = 0.5$. At a churn rate of 0.9, a random node in the 1-hop case is disconnected from the largest connected component with a probability of nearly 75%. However, a node that has trust relationships with her friends of friends has a disconnection probability of about 10%. A more striking finding is when the churn rate is 0.95, a node in the 1-hop network is disconnected most of the time, while a 2-hop node’s disconnection probability is around 20%.

Our result shows that a simple design of an overlay network leveraging trust relationships within a 2-hop neighborhood of each node is significantly more robust to heavy churn than its 1-hop counterpart.

6.3.2 Sybil Attack

Here we study the security implications of the above design, particularly from a Sybil defense perspective. Social network-based Sybil defense schemes [65, 91, 104, 105] leverage trust relationships among honest users for their robustness against Sybil attacks [43]. Their main idea is that creating trust relationships is difficult due to the requirement of high social engineering effort, and hence, the number of trust relationships (referred to as attack edges) that an adversary can form with honest users of the system is limited. The robustness of each scheme relies on the attack edges being few. The more the attack edges, the worse the robustness.

Since the above two-hop neighborhood design motivates creating new trust relationships with friends of friends for better network connectivity under heavy churn, an important question follows: does this design consequently help create new attack edges between honest and Sybil nodes that are obviously undesired? To answer this question, we carry out the following experiment. First, we create a Sybil region denoted by a random graph comprising 10K nodes with an average degree of 5. Second, we assume the Yahoo graph representing the honest region. Third, we create $g$ attack edges between an honest node and a Sybil node,
Figure 6.5: Pros and Cons of exploring two-hop neighborhood.
both being selected uniformly at random. Next, we measure (i) attack edges that may have been created in the 2-hop neighborhood design, and (ii) escape probability: the probability that a random walk starting from an arbitrary honest node traverses an attack edge and escaping into the Sybil region under both 1-hop and 2-hop scenarios.

Figure 6.5(b) plots the attack edges created in the 2-hop network relative to the number of honest nodes as the attack edges in the original 1-hop network increases. The number of attack edges in the 2-hop network is nearly 12 times that of the number of attack edges in the 1-hop network. This result indicates that the above 2-hop neighborhood design has a particularly poor consequence for Sybil defenses. Figure 6.5(c) plots the escape probability versus the attack edges in the 1-hop network, with the length of the random walk set to 15. Although a random walk in the 2-hop network escapes into the Sybil region with a greater probability than in the 1-hop network, we note that the difference in both scenarios is not as high when compared to their respective attack edges. This is perhaps due to a large number of edges created among honest users in the 2-hop network, which diminishes the role of more attack edges created.

6.4 Adaptive 2-Hop Design

In this section, we explore a new design point in the tradeoff between network connectivity and security. We first outline our goal and assumptions for our design. We then describe the rationale of our approach and the details of our method.

6.4.1 Goal and Assumptions

Our main goal in this chapter is to improve the connectivity of a low degree distributed social network under heavy churn, while incurring a marginal security cost. More specifically, the system should have the following desired properties:

- **Network Connectivity**: the network should be able to remain connected with a high probability under churn rates as high as 85% to 95%;

- **Attack Resilience**: the robustness of the network to attacks should not be drastically reduced.

Based on the findings in the above sections, we make the following plausible assumptions that influence our design. First, the social network graph is fast mixing. Second, there is a strong correlation between the number of friends a user has, the amount of time she is online and her user activity. Third, a fraction of users are online contiguously. In other words, despite dynamic nature in the network, some users are online in two successive time intervals during which other nodes may join or leave the network.
6.4.2 Approach Overview

The rationale of our approach is based on two network properties: expansion and conductance. The measure of expansion \[69-71\] captures how well a subgraph or a sample \(S\) is connected to the rest of the network. Improving the connectivity of the network requires adding links to nodes that contribute to the expansion of the node’s neighborhood. The measure of conductance \[64\] captures how well the sample \(S\) is connected within itself relative to the immediate neighborhood of the sample. The robustness of most social network-based trust schemes depend on the lower conductance of the honest region with respect to the rest of the network. In other words, the honest region would form a stronger community within itself, resulting in a sparse cut between the honest- and Sybil regions. Obviously, the assumption is that the number of attack edges is limited.

Low degree nodes which are usually the most vulnerable to disconnection due to churn must seek links from higher expansion nodes to improve their connectivity in the network. High degree nodes, on the other hand, must either refrain from adding more links or seek nodes that decreases the conductance around its neighborhood.

6.4.3 Method

We now describe the details of our approach.

Computing Neighborhood Measures

Given a 1-hop social network, each node computes both expansion and conductance for each of its two-hop neighbors. The computation is performed as follows. A node \(v\) begins with a sample \(S\) containing itself in the beginning. The node \(v\) then adds its immediate 1-hop neighbors into \(S\). The neighborhood \(N(S)\) constitutes \(v\)'s 2-hop neighbors. For each node \(k\) in \(N(S)\), \(v\) computes a score for \(k\)'s expansion contribution \(EC(k) = |N\{k\} - (N(S) \cup S)|\), which is simply the number of new neighbors \(k\) contributes to the expansion of \(v\)'s sample \(S\).

Similarly, the conductance is computed as follows. For each node \(k\) in \(N(S)\), let \(n\) be the number of outgoing edges from the sample \(N(S) \cup \{k\}\). The conductance score for \(k\) is \(n\) divided by the sum of degrees of all nodes in the sample \(N(S) \cup \{k\}\). We assume here that the graph is much larger than a 2-hop neighborhood of any node.

A Two-Hop Network

We now describe how a 2-hop network can be created using the above computed neighborhood measures in a static environment where there are no nodes joining or leaving the network. First, a node \(v\) adds a maximum of \(MAX_{XP\_NBRS}\) 2-hop neighbors with the highest expansion scores. Next, \(v\) adds a maximum of \(MAX_{CON\_NBRS}\) 2-hop neighbors which have a lower conductance score.
neighbors with lowest conductance scores. This ensures that the network overall is better connected through high expansion nodes, as well as robust to attacks due to lowering the conductance of the network. Trust schemes such as social network-based Sybil defenses can then exploit low conductance of the honest region with respect to the whole graph in order to differentiate from the Sybil region.

**Node Join**

We base our node join feature on the topology adaptation scheme that was used to improve the connectivity in the Gia network [32].

When a node $v$ joins the network, it first adds its 1-hop neighbors who are online. Next, $v$ checks if the number of active neighbors in the topology of the overlay network is less than $MIN_{NBRS} = 5$. If the neighborhood is sufficiently populated, $v$ does not add more links. Otherwise, it adds a maximum $MAX_{XP_{NBRS}} = 2$ of its 2-hop neighbors with highest expansion who are online. Such high expansion nodes would improve its connectivity in the network. If the node degree of $v$ is still not greater than $MIN_{NBRS}$, it seeks at most $MAX_{CON_{NBRS}} = 3$ with lowest conductance scores. Seeking low conductance neighbors improves the trust within this neighborhood.

We now describe how a node $u$ which is already present in the network gets a request of adding a link from a potential neighbor $v$. If $v$ is $u$’s immediate neighbor, they both add each other. If $v$ and $u$ are respective 2-hop neighbors, node $u$ first checks if $u$’s limit $MAX_{NBRS} = 10$ is reached. If there are slots remaining, $u$ accepts the request from $v$. Otherwise, $u$ adds $v$ if $v$ lowers the conductance of $u$’s neighborhood. This is because if $u$ has reached its maximum capacity, it means that $u$ is already well connected. In this case, adding nodes with higher expansion would not improve $u$’s connectivity to a great extent. The node in $u$’s list that is replaced $z$ is the one with high expansion. Since $z$ is generally well connected, $z$ can afford to lose one connection.

**Node and Link Failures**

Both node and link failures are determined using symmetric link failure detection [27]. If a node $v$ detects that a neighboring node has failed, it performs similar steps as the above node join operation. That is, if need be, $v$ seeks for more new neighbors preferably with high expansion for improving its network connectivity.

### 6.5 Trace-Driven Evaluation

In this section, we evaluate our method based on the traces of the Yahoo! Messenger network which we described in Section 6.1.
6.5.1 Experimental Setup

Similar to the previous sections, we use the same 100K nodes graph of the Yahoo social network to perform our experiments. We assume the network to exhibit very high churn rates close to 0.9. In other words, nearly 10% of all users of the distributed social network-based system are online at any instance. We set $\gamma = 0.5$ as its default value, which indicates that users with more friends are likely to remain online slightly longer than the ones with fewer friends. We believe this is a reasonable parameter setting based on the observations in Section 6.1.

In such a dynamic environment, we study the connectivity properties of the network in three scenarios: 1-hop, full 2-hop, and adaptive 2-hop. The first scenario, 1-hop, deals with a design that only uses the direct social network relationships to be connected in the trust overlay network. This design is often employed in most social network-based distributed system studies. The second scenario, full 2-hop, enables each node to leverage trust relationships with all nodes within its 2-hop neighborhood to remain connected in the network. The third scenario, adaptive 2-hop, explores only a few trust relationships in the 2-hop neighborhood to improve its connectivity in the network. Neither the second nor the third designs have been explored in detail in previous works.

6.5.2 Tradeoff: Robustness vs Connectivity

We have seen from Section 6.3 that a full 2-hop design improves the network connectivity compared to the 1-hop design but at the cost of creating an order of magnitude more new attack edges. Here we examine how our adaptive 2-hop method fares in terms of both connectivity and robustness. We begin with a simple 1-hop network of the Yahoo graph, which we consider as the honest region. We add a Sybil region of 10K nodes with an average degree of 5 along with 5K attack edges to the honest region, similar to the Section 6.3. Next, each node in this network adds a maximum of $MAX_{NBRS}$ select nodes within its 2-hop neighborhood. The selection process is based on the expansion contribution of a candidate node, i.e., how many new neighbors is the candidate contributing to its 1-hop neighborhood [69–71]. We then measure the number of attack edges in the resultant network as a function of the 2-hop neighbors.

Figure 6.6(a) plots the result comparing the performance of the expansion-based neighbor selection technique with a baseline method choosing 2-hops neighbors at random. The 2-hop attack edges grows significantly with addition of only a few 2-hop neighbors. A mere 25 2-hop neighbors seem to be sufficient to create an order of magnitude more attack edges. Moreover, the expansion-based technique performs worse than the random one. Since the former technique relies on adding those 2-hop neighbors with more new connections, the consequence is that nodes close to the attack edges are vulnerable to exploring a larger segment of the Sybil region. However, when less than 10 2-hops neighbors are added, the
performance of both techniques is comparable.

We now shift our focus to the connectivity of the network as a function of the 2-hop neighbors. In this experiment, we set the churn rate to 0.9. Figure 6.6(b) plots the disconnection probability as more 2-hop neighbors are added. As expected, the more the 2-hop neighbors, the lower the disconnection probability. A more interesting result is that, adding only a few 2-hop neighbors (5 in this case), the probability of a node getting disconnected is 0.2, which is much better than 0.8 in the 1-hop network. Further, the expansion-based neighbor selection technique fares marginally better than the random one.

Figure 6.6(c) compares the performance of random and expansion-based techniques as a function of the churn rate. At high churn rates of 0.9 and above, the expansion-based technique outperforms the random one. The expansion-based technique enables a node with 5 2-hops neighbors in a network under a churn rate of 0.95 is likely to be connected more than half the time, while the disconnection probability of a similar node is nearly 0.8 with the random technique.

From the three figures: Fig. 6.6(a), Fig. 6.6(b), and Fig 6.6(c) we infer that adding a small number of 2-hop neighbors (less than 10 in our case) with expansion-based selection technique makes the network more resilient to heavy churn, at the cost of a modest increase in the number of attack edges in the 2-hop network.

### 6.5.3 Steady State

We now study how the dynamics in the network such as nodes joining and leaving affect the performance of the system. In this experiment, we consider a steady state where the number of nodes that join the network is equal to the number of nodes that leave the network during any time interval. This case is similar to the one we observed in the Section 6.1.3: from 9 AM to 9 PM, there were nearly 15% of all users online in each hourly time interval (Fig. 6.2(a)). Also during this period, about 40% of these online users were common in two consecutive hourly time intervals (Fig. 6.2(b)).

We examine the connectivity of the network during the transition from one time interval to the next during which a few nodes join while a few others leave. Our experiment moves in simulation rounds where each round corresponds to a time interval. We define join-to-leave ratio to be equal to the number of nodes joining the network divided by the number of nodes leaving before the end of a simulation round. In a steady state, the join-to-leave ratio is 1. In our experiment, we consider a range of 20% to 35% of nodes are common in two successive time intervals. We apply our adaptive 2-hop method to handle the transition from one interval to the next. Recall that our method seeks a small number of neighbors in the 2-hop neighborhood that have high expansion. We compare our method to both 1-hop design which existing systems adopt and full 2-hop design which takes into account all nodes in the 2-hop neighborhood.
Figure 6.6: Tradeoff between the connectivity of the network and the number of attack edges in the 2-hop network.
Figure 6.7: Connectivity of the network in a steady state.

Figure 6.7 plots the disconnection probability versus the simulation rounds, in the three different designs. The performance of our adaptive 2-hop method is comparable to the full 2-hop design, while outperforming the 1-hop design. Our method improves the network connectivity as the simulations rounds increase. It is important to note that our method is resilient to even a small fraction of 20% nodes being common in successive simulation rounds.

6.5.4 Flashcrowd

Flashcrowd refers to the scenario where a large number of nodes join the network relative to the ones already in the system within a short period in time. Flashcrowds can be observed in both centralized and distributed systems. Websites hosting content that gains instant popularity attracts vast amount of traffic within a few hours or even minutes. P2P systems such as BitTorrent also notice flashcrowds, particularly when a torrent for a popular content becomes available. In Section 6.1.3 we also noticed a flashcrowd of users coming online in the morning from 5 AM to 10 AM after which the rate of joining and leaving remains stable. The join-to-leave ratio during this flashcrowd was in the range of 1.3 and 1.6.

Handling a growing population is critical to any system design. We now study the connectivity of the network under a flashcrowd. We adopt the simulation setup of the steady state experiment, and apply a flashcrowd effect with a join-to-leave ratio in the range of 1 and 2. We set the fraction of nodes common in consecutive time intervals to 0.25.

Figure 6.8 plots the disconnection probability versus the simulation rounds, varying the join-to-leave ratio. The performance of our adaptive 2-hop method is comparable to the full 2-hop design in the flashcrowd scenario as well. The network connectivity grows with both simulation rounds and higher join-to-leave ratio. This is particularly noticeable in the 1-hop case where high join-to-leave ratio decreases the disconnection probability with time. This is due to more nodes joining the system rather than leaving and forming links with other
6.5.5 Recession

Recession refers to the decline in the number of nodes in the system where more nodes leave than join. Recession can be observed in P2P systems such as BitTorrent where the popularity of a torrent fades, resulting in nodes leaving the swarm. We observed a recession in Section 6.1.3 where a large number of nodes leave during the late night from 11 PM to 5 AM. During this period, the worse join-to-leave ratio was 0.65.

In this experiment, we apply a recession effect on the network with a join-to-leave ratio in the range of 0.5 and 1. Similar to the previous experiment, we adopt its simulation setup, and set the fraction of nodes common in consecutive time intervals to 0.25.

Figure 6.9 plots the disconnection probability versus the simulation rounds, varying the join-to-leave ratio. Our adaptive 2-hop method is seen to be resilient to a recession. The network connectivity remains the same even when double the number of nodes leave the network than the ones that join (join-to-leave ratio = 0.5). However, the connectivity problem exacerbates in the 1-hop design with more nodes leaving than joining.

6.6 Related Work

Here we review the literature in the field of distributed social networks as well as P2P systems dealing with churn.

6.6.1 Decentralized Social Networks

Decentralized social networks offer an alternative to OSNs, where users hold all the ownership of their data and perform secure communication with their friends without passing any
central entity. Diaspora [2] is prime example where users can setup their own servers to host content, form friends, share updates and multimedia content with others. Safebook [39] adopts a peer-to-peer architecture and real-world trust relationships to build a privacy-preserving social network. Peerson [26] is another effort that builds on a peer-to-peer infrastructure. Peerson users keep control of their data through encryption, key management and access control in a decentralized setting.

6.6.2 Social Networks & Distributed Systems

A large body of work leverages social networks by incorporating their properties such as inherent trust relationships among users and graph structure into the designs of social network-based Sybil defenses (SNSD) schemes [40, 65, 84, 91, 92, 104, 105]. Each of these schemes makes two fundamental assumptions. First, although an attacker can create arbitrary number of identities, she cannot establish arbitrary number of trust relationships (attack edges) with honest users since forming a trust relationship requires high social engineering cost. This leads to a sparse cut between Sybil region containing malicious identities and non-Sybil/honest region containing honest users in the graph, which is then exploited by these schemes. Second, a social network graph is expander-like and fast-mixing [76] in that a random walk in the graph quickly reaches a stationary distribution. Hence, a short random walk starting from a node in the non-Sybil region rarely escapes into the Sybil region.

Another line of work investigates how trust relationships can be used for send secure communications. OneSwarm [50] leverages the social network to build an overlay over which data transfer is performed. Since these trust relationships are known only to the endpoints of the edge, the communication can be monitored easily. X-Vine [75] also uses a similar rationale for performing secure lookups. The trust overlay is used for the construction and maintenance of the routing tables inspired by the virtual ring routing mechanism [27]. MCON [93] explored the 2-hop neighborhood for a robust design of routing tables in a trust
overlay controlled by a central entity. However, they did not examine the consequences of adding trust relationships from 2-hop neighbors.

None of these studies however have examined heavy churn conditions. Also, the social networks graphs in their evaluation has a much higher average degree than the one observed in the Yahoo social network graph.

6.6.3 P2P Churn

Churn has often been considered as a key challenge in developing peer-to-peer systems. Stutzbach and Rejaie [90] measured churn in three peer-to-peer file sharing systems (BitTorrent, Kad and Gnutella). Their results corroborate and extend previous efforts to show that peer uptime is skewed in all three systems, with many users staying online only for a few minutes at a time. This observation, in turn, has direct implications for the connectivity of the system. However, Stutzbach and Rejaie note that previous uptime is a reasonable predictor of future uptime for peers, a property that can be leveraged for connectivity resilience. Our work complements Stutzbach and Rejaie’s and similar efforts in quantifying churn in file sharing systems by considering a distributed social network, and also by noting properties related to peer uptime in this context that can be leveraged for improving system resilience.

On a different perspective, Rhea et al. [85] present an overview of the literature on the measurement of session durations in file-sharing p2p systems. This literature points to median session durations in the order of few minutes to one hour. Moreover, Bhagwan et al. [22] measured Overnet to find that users are online on a median of 30% of the period that comprises their participation in the system. Rhea et al. show how the assumptions related to churn drastically affect the performance of DHT systems, requiring their mechanisms to be developed with churn as a primary concern. Our work aims at contributing to the conception of realistic churn assumptions for the development of distributed social network based solutions.

6.7 Conclusion

We analyzed an operational distributed social network – Yahoo! Messenger – to derive design considerations for building a practical social network-based distributed system. We observe that the Yahoo’s social network graph is fast mixing, making it suitable for building social network-based Sybil defense applications such as SybilLimit [104] and Whanau [65] among others [75,91,105]. Our study also reveals a heavy churn with only 5%-15% of all nodes in the network being online at any moment. We show that trust overlays built on top the social network under such churn disintegrates into multiple disconnected components.
A naive design extending the trust relationships to 2-hop neighbors increases the network connectivity at the cost of being more vulnerable to attacks.

We explore this design point in the tradeoff between network connectivity and attack resilience. Our proposed adaptive 2-hop outperforms the 1-hop solution in terms of network connectivity while having a comparable performance with the naive 2-hop approach. In terms of attack resilience, our method fares significantly better than the full 2-hop design and only modestly worse than the 1-hop design. Our method is seen to be resilient under steady state, flashcrowd, recession and heavy churn.

Our analysis of an operational distributed social network and the proposed adaptive 2-hop design can help motivate practical designs for future social network-based distributed systems.
Chapter 7

Conclusion

In this thesis, we examined the effectiveness of online social media (OSM) under trying conditions. We leveraged the network structure of OSM to devise scalable and effective techniques that are resilient even in adversarial environments. Also, we made a strong case for the use of distrust to defend against various strategic attacks in multiple scenarios. Moreover, we highlighted some of the limitations in existing efforts to build robust and secure distributed social networks. We explored a new design point in this field which may motivate practical designs for future social network-based distributed systems.

In this chapter, we present our conclusions answering the research questions in the introduction of the thesis, and throw light on a potential road ahead.

7.1 Conclusions

Here we outline our main conclusions in the thesis:

1. Recommending items in a large and sparse dataset of OSM is non-trivial. On our crawled Flickr dataset containing over 120K users and 80K photos favorited by them, our adaptations of scalable link prediction algorithms outperform a traditional collaborative filtering technique. We found that algorithms recommending items within a 3-hop distance in the user-item graph outperform others, suggesting that users are mostly interested within a small proximity of their tastes in the user-item space.

2. We verified our hypothesis that a family of schemes based on random walk with restart from pre-trusted nodes is vulnerable to targeted attacks by examining a renowned family member – EigenTrust (ET) – under a novel attack based on eigenvector centrality. We propose Personalized EigenTrust (PET) which (i) enables each user to choose her trusted nodes from the social network, thereby eliminating the need of pre-trusted nodes and making the system self-sufficient, (ii) is effective in networks operating under various transaction models based on distributions such as random, community-like
and power-law, and (iii) is robust to many types of attacks including the targeted one based on eigenvector centrality. Our simulation results reveal that PET outperforms ET under diverse transaction models and attack strategies.

3. Distrust in the form of negative feedback can play a vital role in negating Sybils from outvoting honest users in OSM. By modeling trust and distrust relationships as a signed network, we verified our hypothesis that attack edges constrain the paths along positive edges between two endpoints of a defense edge (i.e., a distrust relationship between honest and Sybil nodes). Based on this insight, we reduce the voting capability of Sybils by 80% with a defense edge each from a mere 25% of all honest users. Our method is also resilient to ‘friendly fire’, where honest users mistake each other for Sybils. Moreover, we devised a novel pathfinder algorithm that can find a path between any two nodes in a 1 million-node expander graph within a millisecond on a commodity laptop.

4. Distrust can also be useful in identifying Sybils even under targeted attacks. We showed how current Sybil detection schemes including the state-of-the-art approach, SybilRank, perform worse than naive technique under an attack based on eigenvector centrality. We showed that the robustness of SybilRank under such an attack can be improved with minimal changes by incorporating our novel generic framework built on trust and distrust. The results show that, with one defense edge each from an honest node, the robustness of our method is near-optimal. Our method is also resilient to counter attack edges where Sybils initiate negative edges toward honest users.

5. Our analysis of Yahoo! Messenger, an operational distributed social network, highlights practical limitations for existing social network-based distributed systems. We observed that the Yahoo clients exhibit a churn rate of 85% at best to 95% at worst. At such high churn rates, we showed that the trust overlay which is fundamental to the functioning of such systems disintegrates into multiple disconnected components. Furthermore, we showed that exploring 2-hop neighborhood increases the connectivity under heavy churn which comes at the cost of an order of magnitude in the attack capability of an adversary. We proposed an adaptive 2-hop design which explores a new design point in the tradeoff between network connectivity and attack resilience. The results from our trace-driven experiments reveals that the adaptive 2-hop method fills an important point in the design space making it significantly more robust than 1-hop solution and with a modest attack resilience.
7.2 Future Work

Various insights from the thesis may open a window of new opportunities for future researchers:

1. The low values of precision and recall in our work suggest that recommending items in OSM is highly challenging, and hence requires significant attention from IR research community. Studying link prediction algorithms on a wide variety of OSM such as Digg, YouTube, and Flickr would help understand how the network structure and the dynamics impact the performance of these algorithms. Insights from this analysis may help in designing better recommender algorithms. Another factor to consider is temporal dimension. Since popularity of items often have very short lifetimes in OSM, recommender algorithms should take into account age of items to improve their performance. It will also be interesting to see the impact of photo recommendations (using the above techniques or otherwise) in Flickr, since Flickr does not offer explicit recommendations unlike YouTube.

2. Future trust schemes should evaluate their robustness under the centrality attack we devised, particularly since our experiments showed a large negative impact on highly popular reputation systems (EigenTrust) and Sybil defenses (SybilRank). While we discussed attack strategy based on eigenvector centrality, analyzing the impact of various centrality measures such as betweenness, closeness and alpha can equally be insightful. Does an attack strategy based on edge betweenness centrality would be more harmful on a network with community structure than an eigenvector centrality strategy? What network structure requires an adversary to adopt which attack strategy? What would be a good defense strategy for a given network structure against various centrality attacks?

3. The role of distrust in defense schemes should be studied in more detail from various perspectives. Sociological and psychological understanding of distrust in adversarial environments would help design better defense algorithms that can work in the wild. How do we design incentives for honest users to report malicious behavior? How do we design disincentives for an adversary without honest users’ participation? Distrust, unlike trust, may be less appealing for users to accept in public. How do we design privacy-preserving mechanisms that anonymize the source of negative feedback without reducing the effectiveness of a defense scheme?

4. Future designers of social network-based distributed systems should take into account churn as a major design factor. Furthermore, existing systems require a radical redesign considering heavy churn in the network. One line of research in this field is to revisit a prominent application: performing lookups on top of the trust overlay
network [65], More specifically, is it feasible to design a trust overlay that has a low query failure rate under a high churn rate with a reasonable network overhead? Another line of research may explore designing incentives for low degree nodes to participate in the network, thereby increasing the user contribution as a whole as well as reducing the load on high degree nodes.
Bibliography

[10] Twitter may have 500m+ users but only 170m are active, 75clients. http://techcrunch.com/2012/07/31/twitter-may-have-500m-users-but-only-170m-are-active-75-on-twitters-own-clients/.


[22] Ranjita Bhagwan, Stefan Savage, and Geoffrey Voelker. IPTPS ’03.


the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM ’06, pages 351–362, New York, NY, USA, 2006. ACM.


[41] Cristobald De Kerchove and Paul Van Dooren. The pagetrust algorithm: How to rank web pages when negative links are allowed? SDM ’08.


Summary

Securing Social Media: A Network Structure Approach

Online social media (OSM) such as Facebook, YouTube, and Flickr enable millions of users to publish and share millions of messages, videos, and photos with others. Such a vast user base and the wealth of content in OSMs, however, present various challenges. In this thesis, we address some of these challenges such as scalability, adversarial behavior of users, and decentralization capabilities by leveraging the graph structure of the social network among users.

In Chapter 2, we study the problem of recommending relevant content to individual users from the large repositories available in OSMs. We model the friendships among users and users’ content preferences by a user-item graph. On this graph, we use adapted graph-based link prediction algorithms to recommend items to users in a personalized fashion. Our experiments on Flickr show that our adaptations of the link prediction algorithms outperform the classical item-based collaborative filtering techniques.

In Chapter 3, we investigate the shortcomings of EigenTrust (ET, which is a renowned global reputation system. We identify that ET is vulnerable to community structure and centrality attacks targeted towards ET’s pre-trusted nodes. To alleviate these concerns, we propose Personalized EigenTrust (PET) which enables each user to choose her own trusted users from the social network, thereby eliminating the need of system-wide pre-trusted users and making the system self-sufficient. Our simulation results reveal that PET outperforms ET under diverse transaction models and attack strategies.

In Chapter 4, we examine the problem of voting in OSM under Sybil attacks. In Sybil attacks an adversary user uses multiple cheap identities to outvote honest users. We propose a mechanism to minimize the votes from Sybil identities by leveraging (i) trust which is inherent among users in OSMs, and (ii) distrust between honest users, who identify some of the spam content items and the Sybil identities who promoted them. Our simulation results on datasets of popular OSM show both the feasibility and the efficiency of incorporating distrust alongside trust to defend against Sybil attacks. Our method outperforms the state-of-the-art approach, SumUp.

In Chapter 5, we focus on improving SybilRank. SybilRank is the state-of-the-art social network-based Sybil detection scheme that uses the available distrust relationships in the system. We first build a lightweight framework based on trust and distrust relationships and
then adapt SybilRank to incorporate this framework with minimal changes. Our experiments on popular OSM datasets show that our method significantly outperforms SybilRank even under targeted attacks.

In Chapter 6, we explore the feasibility of building distributed social network-based systems such as OneSwarm and Whanau that are practical in the real world. We note that the effectiveness of such systems relies on a connected overlay network built using the trust relationships among users. Under realistic churn, we find that the trust overlay network disintegrates into multiple connected components that essentially make such a system impractical in the wild. We show that the 2-hop neighborhood of each user can be explored to improve the connectivity, however at the cost of making the system more vulnerable to Sybil attack. We propose an adaptive 2-hop design that improves the connectivity of nodes as well as their resilience to attacks by forming relationships with only select 2-hop neighbors. Our trace-driven experiments under trying conditions reveal that our adaptive 2-hop method fills an important point in this design space making it significantly more resilient to churn than 1-hop solutions, only at the cost of a modest increase in the attack capability of the adversary.
Samenvatting

Het beveiligen van Sociale Media: Een Netwerk Structuur Benadering

Online sociale media (OSM), zoals Facebook, YouTube en Flickr, stellen miljoenen gebruikers in staat om miljoenen berichten, video’s en foto’s te publiceren en met anderen te delen. Echter een dergelijke grote gebruikersgemeenschap en de enorme hoeveelheid data in OSMs geven verschillende uitdagingen. Dit proefschrift richt zich op een aantal van deze uitdagingen, zoals schaalbaarheid, conflicterend gedrag van gebruikers en decentralisatie mogelijkheden door gebruik te maken van de inherente verbindingsstructuur tussen gebruikers van het sociale netwerk.

In hoofdstuk 2 wordt het probleem van het aanbevelen van relevante data aan individuele gebruikers uit de grote hoeveelheid aanwezige data in OSM’s bestudeerd. Hiertoe worden de vriendschappen tussen gebruikers en de datavoorkeuren van de gebruikers in een gebruikers-data item graaf gemodelleerd. Door gebruik te maken van een aangepast en al eerder effectief gebleken link predictie algoritme kunnen we gebruikers op een gepersonaliseerde manier data items aanbevelen. Experimenten met Flickr laat zien dat dit aangepaste algoritme beter werkt dan het klassieke op items gebaseerde collaboratieve filter algoritme.

In hoofdstuk 3 worden de tekortkomingen van EigenTrust (ET) onderzocht, dat wereldwijd gebruikt wordt als reputatie systeem. Aangetoond wordt dat ET kwetsbaar is voor de structuur van een gemeenschap en op centraliteitsaanvallen gericht op door ET vertrouwde knooppunten. Om deze bezwaren weg te nemen, is het Gepersonaliseerde EigenTrust (PET) algoritme ontwikkeld, waarmee elke gebruiker zijn eigen vertrouwde gebruikers kan kiezen uit het sociale netwerk, waardoor de noodzaak van systeem brede vertrouwde gebruikers wordt gelimineerd en daarmee het systeem zelfvoorzienend wordt. Simulatieresultaten tonen aan dat PET ET overtreft onder diverse transactie modellen en aanval strategieën.

In hoofdstuk 4 wordt het probleem van het uitbrengen van stemmen in OSM’s onderzocht terwijl het OSM onderhevig is aan Sybil aanvallen. Bij Sybil aanvallen wordt gebruik gemaakt van kwaadwillende gebruikers die gebruik maken van meerdere goedkope identiteiten om eerlijke gebruikers te overstemmen. Een mechanisme wordt voorgesteld om de stemmen van Sybil identiteiten te minimaliseren door gebruik te maken van (i) het vertrouwen dat inherent tussen gebruikers in het sociale netwerk onder de gebruikers in OSM’s en (ii) het wantrouwen tussen eerlijke gebruikers, die een deel van de spam content items identificeren en de Sybil identiteiten die ze aanbevelen. Simulatieresultaten op
datasets van populaire OSM’s tonen zowel de haalbaarheid als de efficientie aan van het opnemen van wantrouwen naast vertrouwen tegen Sybil aanvallen. De gepresenteerde methode werkt beter dan de state-of-the-art aanpak SumUp.

Hoofdstuk 5 is gericht op het verbeteren van SybilRank. SybilRank is het state-of-the-art Sybil detectieschema voor OSM’s dat gebruik maakt van de aanwezige wantrouwrelaties in het systeem. Daartoe is een lichtgewicht framework ontworpen dat is gebaseerd op vertrouw- en wantrouwtrelaties. Vervolgens is SybilRank met minimale aanpassingen geschikt gemaakt voor dit framework. Experimenten op populaire OSM datasets laten zien dat de aangepaste methode aanzienlijk beter presteert dan SybilRank, zelfs onder gerichte aanvallen.

In hoofdstuk 6 wordt de praktische haalbaarheid van het bouwen van gedistribueerde sociale netwerk gebaseerde systemen zoals OneSwarm en Whanau onderzocht. De effectiviteit van dergelijke systemen is gebaseerd op een verbonden overlay netwerk opgebouwd met behulp van de vertrouwensrelaties tussen gebruikers. Onder realistische churn omstandigheden, blijkt dat het vertrouwen overlay netwerk uiteen valt in meer dan n, onderling niet verbonden overlays die in wezen een dergelijk systeem onpraktisch maakt. Aangetoond wordt dat de 2-hop omgeving van elke gebruiker kan worden gebruikt om de connectiviteit van de knopen te verbeteren, dit echter ten koste van de kwetsbaarheid voor aanvallen van Sybils. Een aangepast 2-hop ontwerp wordt gepresenteerd dat de connectiviteit van de knopen verbetert alsmede hun weerstand tegen aanvallen door het aangaan van geselecteerde relaties met 2-hop buren. Trace-driven experimenten onder uitdagende omstandigheden laten zien dat deze adaptieve 2-hop methode een belangrijk punt in de ontwerp ruimte realiseert, waardoor het systeem aanzienlijk beter bestand is tegen churn dan 1-hop oplossingen. Dit ten koste van een bescheiden verhoging van de aanvalsmogelijkheden van tegenstanders.
Curriculum vitae

Nitin Chiluka was born and brought up in the southern city of Hyderabad, India. He moved to the northern part of the country for his dual degree – Bachelor’s and Master’s – in Information Technology at ABV - Indian Institute of Information Technology and Management, Gwalior, India, where he graduated in 2007. In January 2009, he began as a PhD candidate in Computer Science at Delft University of Technology under the guidance of Prof. Henk Sips and Dr. Johan Pouwelse. During his PhD, he primarily worked in the field of social networks and peer-to-peer systems. For three academic years beginning 2010, he was a lab assistant for the ‘Parallel Algorithms and Parallel Computers’ course taught by Prof. Henk Sips and Prof. Cees Witteveen. In early 2012, he worked as a visiting researcher at Universidade Federal de Campina Grande (UFCG) in Brazil under Prof. Nazareno Andrade.

Papers:


