

MailRank: Using Ranking for Spam Detection

Paul - Alexandru Chirita

L3S Research Center /
University of Hannover
Deutscher Pavillon
Expo Plaza 1
30539 Hannover, Germany
chirita@l3s.de

Jörg Diederich

L3S Research Center /
University of Hannover
Deutscher Pavillon
Expo Plaza 1
30539 Hannover, Germany
diederich@l3s.de

Wolfgang Nejdl

L3S Research Center /
University of Hannover
Deutscher Pavillon
Expo Plaza 1
30539 Hannover, Germany
nejdl@l3s.de

ABSTRACT

Can we use social networks to combat spam? This paper investigates the feasibility of MailRank, a new email ranking and classification scheme exploiting the social communication network created via email interactions. The underlying email network data is collected from the email contacts of all MailRank users and updated automatically based on their email activities to achieve an easy maintenance. MailRank is used to rate the sender address of arriving emails such that emails from trustworthy senders can be ranked and classified as spam or non-spam. The paper presents two variants: Basic MailRank computes a global reputation score for each email address, whereas in Personalized MailRank the score of each email address is different for each MailRank user. The evaluation shows that MailRank is highly resistant against spammer attacks, which obviously have to be considered right from the beginning in such an application scenario. MailRank also performs well even for rather sparse networks, i.e., where only a small set of peers actually take part in the ranking of email addresses.

Categories and Subject Descriptors

G.2.2 [Discrete Mathematics]: Graph Theory; H.3.4 [Information Systems]: Information Storage and Retrieval—*Systems and Software*; H.2.7 [Information Systems]: Database Management—*Security, Integrity and Protection*

General Terms

Algorithms, Experimentation, Measurements

Keywords

Email Reputation, SPAM, MailRank, Personalization

1. INTRODUCTION

While scientific collaboration without email is almost unthinkable, the tremendous increase of unsolicited email (spam) over the past years [5] has rendered email communication without spam

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'05, October 31–November 5, 2005, Bremen, Germany.
Copyright 2005 ACM 1-59593-140-6/05/0010 ...\$5.00.

filtering almost impossible. Currently, spam emails already outnumber non-spam ones, so-called 'ham emails'. Existing spam filters such as the SpamAssassin System¹, SpamBouncer², or Mozilla Junk Mail Control³ still exhibit some problems, which can be classified in two main categories:

1. **Maintenance**, for both the initialization and the adaptation of the filter during operation, since all spam filters rely on a certain amount of input data to be maintained: Content-based filters require keywords and rules for spam recognition, blacklists have to be populated with IP addresses from known spammers, and Bayesian filters need a training set of spam / ham messages. This input data has to be created when the filter is used first (the 'cold-start' problem), and it also has to be adapted continuously to counter attacks of spammers [7, 19].
2. **Residual error rates**, since current spam filters cannot eliminate the spam problem completely. First, a non-negligible number of spam emails still reaches the end user, so-called false negatives. Second, some ham messages are discarded because the anti-spam system considers them as spam. Such false positives are especially annoying if the sender of the email is from the recipient's community and thus already known to the user, or at least known by somebody else the user knows directly. Therefore, there is a high probability that an email received from somebody within the social network of the receiver is a ham message. This implies that a social network formed by email communication can be used as a strong foundation for spam detection.

Even if there existed a perfect anti-spam system, an additional problem would arise for high-volume email users, some of which simply get too many ham emails. In these cases, an automated support for email ranking would be highly desirable. Reputation algorithms are useful in this scenario, because they provide a rating for each email address, which can subsequently be used to sort incoming emails. Such ratings can be gained in two ways, globally or personally. The main idea of a global scheme is that people share their personal ratings such that a single global rating (called reputation) can be inferred for each email address. The implementation of such a scheme can, for example, be based on network reputation algorithms [6] or on collaborative filtering techniques [17]. In case of a personalized scheme, the ratings (called trust in this case) are typically different for each email user and depend on her personal social network. Such a scheme is reasonable since some

¹<http://spamassassin.apache.org>

²<http://www.spambouncer.org>

³<http://www.mozilla.org/start/1.5/extra/using-junk-control.html>

people with a presumably high global reputation (e.g., Linus Torvalds) might not be very important in the personal context of a user, compared to other persons (e.g., the project manager).

In this paper we propose MailRank, a new approach to ranking and classifying emails according to the address of email senders. The central procedure is to collect data about trusted email addresses from different sources and to create a graph for the social network, derived from each user’s communication circle [1]. There are two MailRank variants, which both apply a power-iteration algorithm on the email network graph: Basic MailRank results in a global reputation for each known email address, and Personalized MailRank computes a personalized trust value. MailRank allows to classify email addresses into ‘spammer address’ and ‘non-spammer address’ and additionally to determine the relative rank of an email address with respect to other email addresses. This paper analyzes the performance of MailRank under several scenarios, including sparse networks, and shows its resilience against spammer attacks.

The paper is organized as follows: Section 2 provides information about existing anti-spam approaches, trust and reputation algorithms, as well as a description of PageRank and some approaches to personalizing it. In Sect. 3, we describe our proposed variants of MailRank, which we then evaluate in Sect. 4. Finally, our results are summarized in Sect. 5.

2. BACKGROUND AND RELATED WORK

2.1 Anti-Spam Approaches

Because of the high relevance of the spam problem, many attempts to counter spam have been started in the past, including some law initiatives. Technical anti-spam approaches comprise one or several of the following basic approaches [16]:

- Content-based approaches
- Header-based approaches
- Protocol-based approaches
- Approaches based on sender authentication
- Approaches based on social networks

Content-based approaches [7] analyze the subject of an email or the email body for certain keywords (statically provided or dynamically learned using a Bayesian filter) or patterns that are typical for spam emails (e.g., URLs with numeric IP addresses in the email body). The advantage of content-based schemes is their ability to filter quite a high number of spam messages. For example, SpamAssassin can recognize 97% of the spam if an appropriately trained Bayesian filter is used together with the available static rules [10]. The main drawback is that they (e.g., the set of static keywords) have to be adapted continuously since otherwise the high spam recognition rate will decrease [10].

Header-based approaches examine the headers of email messages to detect spam. *Whitelist schemes* collect all email addresses of known non-spammers in a whitelist to decrease the number of false positives from content-based schemes. In contrast, *blacklist schemes* store the IP addresses (email addresses can be forged easily) of all known spammers and refuse to accept emails from them. A manual creation of such lists is typically highly accurate but puts quite a high burden on the user to maintain it. PGP key servers could be considered a manually created global whitelist. An automatic creation can be realized, for instance based on previous results of a content-based filter as is done with so-called *autowhitelists* in SpamAssassin. Both blacklists and whitelists are rather difficult to maintain, especially when faced with attacks from spammers who want to get their email addresses on the list (whitelist) or off the list (blacklist).

Protocol-based approaches propose changes to the underlying email protocol. *Challenge-response schemes* [16] require a manual effort to send the first email to a particular recipient. For example, the sender has to go to a certain web page and activate the email manually, which might involve answering a simple question (such as solving a simple mathematical equation). Afterwards, the sender will be added to the recipient’s whitelist such that further emails can be sent without the activation procedure. The activation task is considered too complex for spammers, who usually try to send millions of spam emails at once. An automatic scheme is used in the *greylisting* approach⁴, where the receiving email server requires each unknown sending email server to resend the email again later.

To prevent spammers from forging their identity (and allow for tracking them), several approaches for **sender authentication** [5] have been proposed. They basically add another entry to the DNS server, which announces the designated email servers for a particular domain. A server can use a reverse lookup to verify if a received email actually came from one of these email servers. Sender authentication is a requirement for whitelist approaches since otherwise spammers can just use well-known email addresses in the ‘From:’ line. Though it is already implemented by large email providers (e.g., AOL, Yahoo), it also requires further mechanisms, such as a blacklist or a whitelist, for an effective spam filtering since spammers can easily set up their own domains and DNS servers.

Recent approaches have started to exploit information from social networks for spam detection. Such **social network based approaches** construct a graph, whose vertices represent email addresses. A directed edge is added between two nodes A and B , if A has sent an email to B . Boykin and Roychowdhury [1] initially classify email addresses based on the clustering coefficient of the graph subcomponent: For spammers, this coefficient is very low because they typically do not exchange emails with each other. In contrast, the clustering coefficient of the subgraph representing the actual social network of a non-spammer (colleagues, friends, etc.) is rather high. The scheme can classify 53% of the emails correctly as ham or spam, leaving the remaining emails for further examination by other approaches. Spammers can attack the scheme by cooperating and building their own social networks. Golbeck and Hendler propose another scheme to rank email addresses, based on exchange of reputation values [6]. The main problem of this approach is that its attack resilience has not been verified.

2.2 Trust and Reputation Algorithms

Trust and reputation algorithms have become increasingly popular to rank a set of items, such as web pages (web reputation) or people (social reputation), for example, when selling products in online auctions. Their main advantage is that most of them are designed for high attack resilience.

Web reputation schemes result in a single score for each Web page. PageRank [15] computes these scores by means of link analysis, i.e., based on the graph inferred from the link structure of the Web. The main idea is that “a page has a high rank if the sum of the ranks of its backlinks is high”. Given a page p , the set of its input links $I(p)$ and output links $O(p)$, the PageRank score is computed according to the formula:

$$PR(p) = c \cdot \sum_{q \in I(p)} \frac{PR(q)}{\|O(q)\|} + (1 - c) \cdot E(p) \quad (1)$$

The damping factor $c < 1$ (usually 0.85) is necessary to guarantee convergence and to limit the effect of rank sinks, one very simple attack on PageRank. Intuitively, a random surfer will fol-

⁴<http://projects.puremagic.com/greylisting/>

low an outgoing link from the current page with probability c or will get bored and select a random page with probability $(1 - c)$ (i.e., the \mathbf{E} vector has all entries equal to $1/N$, where N is the number of pages in the Web graph). To achieve personalization, the random surfer must be redirected towards the preferred pages by modifying the entries of \mathbf{E} . Several distributions for this vector have been proposed since: TrustRank [8] biases towards a set of ham pages in order to identify Web spam, HubRank [2] gives an additional importance to hubs, pages collecting links to many other important pages on the Web, etc.

Personalized PageRank [11] uses a new approach: it focuses on user profiles. One Personalized PageRank Vector (PPV) is computed for each user. The personalization aspect stems from a *set of hubs* (H), each user having to select her *preferred pages* from it. For each page of H , an auxiliary PPV called *basis vector* is precomputed. Then, PPVs for any preference set P are expressed as a linear combination of basis vectors. To avoid the massive storage resources the basis hub vectors would use, they are decomposed into partial vectors (encoding the part unique to each page, computed at run-time) and the hubs skeleton (capturing the interrelationships among hub vectors, stored off-line). Section 3.3 discusses how this can be adapted to our email ranking and classification scenario.

Social reputation schemes are usually designed for use within P2P networks. However, they provide an useful insight into utilizing link analysis to construct reputation systems, as well as into identifying different attack scenarios. [21] presents a categorization of trust metrics, as well as a fixed-point personalized trust algorithm inspired by spreading activation models. It can be viewed as an application of PageRank on a sub-graph of the social network. [18] builds a Web of trust asking each user to maintain trust values on a small number of other users. The algorithm presented is also based on a power iteration, but designed for the context of the Semantic Web, composed of logical assertions. Finally, EigenTrust [12] is a pure fixed-point PageRank-like distributed computation of reputation values for P2P environments. This algorithm is also used in the MailTrust approach [13], a collaborative spam filtering approach similar to Vipul's razor⁵. MailTrust weights the spam classifications received from collaborators by a PageRank-like reputation value to minimize the influence of spammers and malicious peers on the collaborative spam filtering scheme.

3. MAILRANK

In order to compute a rank for each email address, MailRank collects data about the social networks derived from email communication of all MailRank users and aggregates them into a single *email network*. Figure 1 depicts an example email network graph. Node U_1 represents the email address of U_1 , node U_2 the email ad-

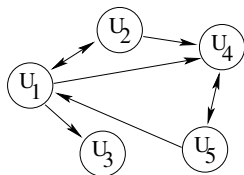


Figure 1: Sample email network

dress of U_2 , and so on. U_1 has sent emails to U_2 , U_4 , and U_3 ; U_2 has sent emails to U_1 and U_4 , etc. These communication acts are then interpreted as trust *votes*, e.g., from U_1 towards U_2 , U_4 and U_3 , and depicted in the figure using arrows.

⁵<http://razor.sourceforge.net>

Building upon the email network graph, we can use a power iteration algorithm to compute a *score* for each email address. This can subsequently be used for at least two purposes, namely: (1) Classification into spam and ham emails, and (2) build up a ranking among the remaining ham emails.

The computation includes the email addresses of all voters (i.e., the ‘actively participating’ MailRank users) and the email addresses specified in the votes. Therefore, it is not necessary that all email users participate in MailRank to benefit from it: For example, U_3 does not specify any vote but still receives a vote from U_1 and will, thus, achieve some score (if U_1 is not a spammer itself).

MailRank has the following advantages:

- **Shorter individual cold-start phase.** If a MailRank user does not know an email address X , MailRank can provide a rank for X as long as at least another MailRank user has provided information about it. Thus, the so-called “cold-start” phase, i.e., the time a system has to learn until it becomes functional, is reduced: While most successful anti-spam approaches (e.g., Bayesian filters) have to be trained for each single user (in case of an individual filter) or a group of users (for example, in case of a company-wide filter), MailRank requires only a single global cold start phase when the system is bootstrapped. In this sense it is similar to globally managed whitelists, but it requires less administrative efforts to manage the list and it can additionally provide information about “how good” an email address is, and not only a classification into “good” or “bad”.
- **High attack resilience.** MailRank is based on a power iteration algorithm, which is typically highly resistant against attacks. This will be discussed for MailRank in particular in Section 4.3.
- **Partial participation.** Building on the power-law nature of email networks, MailRank can compute a rank for a high number of email addresses even if only a subset of email users actively participates in MailRank.
- **Stable results.** Social networks are typically very stable, so the computed ratings of the email addresses will also change only slowly over time. Hence, spammers need to behave well for quite some time to achieve a high rank. Though this cannot resolve the spam problem entirely (in the worst case, a spammer could, for example, buy email addresses from people who have behaved well for some time), it will increase the cost for using new email addresses.
- **Can reduce load on email servers.** Email servers don’t have to process the email body to detect spam. This significantly reduces the computational power for spam detection compared to, for example, content-based approaches or collaborative filters [13].
- **Personalization.** In contrast to spam classification schemes that distinguish only between ‘spam’ and ‘non-spam’, ranking approaches more easily enable personalization features. This is important since there are certain email addresses (e.g., newsletters), which some people consider to be spammers while others don’t. To deal with such cases, a MailRank user can herself decide about the score threshold, below which all email addresses are considered spammers. Moreover, she could use two thresholds to determine ‘spammers’, ‘don’t know’, and ‘non-spammers’. Furthermore, she might want to give more importance to her relatives or to her manager, than to other unrelated persons with a globally high reputation (e.g., Linus Torvalds).
- **Scalable computation.** Power iteration algorithms have been

shown to be computationally feasible even for very large graphs even in the presence of personalization [11].

- **Can also counter other forms of spam.** When receiving spam phone calls (SPIT⁶), for example, one cannot analyze their content before accepting / rejecting them. At best, only the caller identifier is available, which is similar to the sender email address. MailRank can be used to analyze the caller identifier to decide whether a caller is a spammer or not.

The following sections provide more information about each central aspect of MailRank: what data are used by the algorithm, where these data are stored, how the ranks are generated and how we can use them for computing global or personalized reputation scores.

3.1 Bootstrapping the email network

As for all trust / reputation algorithms, it is necessary to collect as many personal votes as possible in order to compute relevant ratings. Collecting the personal ratings should require few or no manual user interactions in order to achieve a high acceptance of the system. Similarly, the system should be maintained with little or no effort at all, thus having the rating of each email address computed automatically.

To achieve these goals, we use already existing data inferred from the communication dynamics, i.e., who has exchanged emails with whom. This results in a global email social network. We distinguish three information sources as best serving our purposes:

1. **Email Address Books.** If a person A has the addresses B_1, B_2, \dots, B_n in its Address Book, then A can be considered to trust them all, or to vote for them.
2. The **‘To:’ Fields** of outgoing emails (i.e., ‘To:’, ‘Cc:’ and ‘Bcc:’). If A sends emails to B , then it can be regarded as trusting B , or voting for B . This input data is typically very accurate since it is manually selected (i.e., it does not contain spammer addresses), and it is more accurate than data from address books, since address books can comprise old or outdated information and there is normally no information available about when the address book entry was created / modified last. Furthermore, address books are private and would have to be released manually by the owner to be accessible for the MailRank system. In contrast, data based on the ‘To:’ fields can also be extracted automatically via a light-weight email proxy deployable on any machine.
3. **Autowhitelists** created by anti-spam tools (e.g., SpamAssassin) contain a list of all email addresses from which emails have been received recently, plus one score for each email address which determines if mainly spam or ham emails have been received from the associated email address. All email addresses with a high score can be regarded as being trusted.

3.2 Basic MailRank

The main goal of MailRank is to assign a rank to each email address known to the system and to use this rank (1) to decide whether each email is coming from a spammer or not, and (2) to build up a ranking among the filtered non-spam emails. Its basic version comprises two main steps:

1. Determine a set of email addresses with a very high reputation in the social network.
2. Apply the power iteration algorithm to the email network graph, biased on the above determined set to compute the final MailRank score for each email address.

Regarding the attack resilience, it is important for the biasing

⁶Spam over Internet Telephony, http://www.infoworld.com/article/04/09/07/HNspamspit_1.html

set not to include any spammer. This is a very efficient way to counter malicious collectives of spammers trying to attack the ranking system [8, 12]. In principle, there are three possible methods to determine the biasing set: manually, automatically, or semi-automatically. A manual selection guarantees that no spammers will be in the biasing set and can in this way counter malicious collectives entirely. An automatic selection can avoid the (possibly costly) manual selection of the biasing set. A semi-automatic selection of the biasing set can use the above described automatic selection to propose a biasing set for being verified manually to be free of spammers. We propose the following heuristics to determine the biasing set automatically:

We first determine the size p of the biasing set by adding the ranks of the R nodes with the highest rank such that the sum of the ranks of these R nodes is equal to 20% of the total rank in the system. Also, we additionally limit p to the minimum of R and 0.25% of the total number of email addresses in the graph⁷. In this manner we limit the biasing set to the few most reputable members of the social network, because of the power-law distribution of email addresses [4, 9]. Thus, we can exclude spammers effectively even if the spammer email addresses constitute the majority in the graph.

The result of the overall MailRank algorithm, the final vector of MailRank scores, can be used to tag an incoming email on the email proxy as (1) non-spammer, if the final score of the sender email address is larger than a threshold T , (2) spammer, if the final score of the sender email address is smaller than T , or (3) unknown, if the email address is not yet known to the system⁸.

Each user can adjust T according to her preferred filtering level. If $T = 0$, the algorithm is effectively used to compute the transitive closure of the email network graph starting from the biasing set. This is sufficient to detect all those spammers for which no user reachable from the biasing set has issued a vote. With $T > 0$, it becomes possible to detect spammers even if some non-spammers vote for spammers (e.g., because the computer of a non-spammer is infected by a virus). However, in this case some non-spammers with a very low rank are at risk of being counted as spammers.

The Basic MailRank algorithm is summarized in Alg. 3.1.

3.3 MailRank with Personalization

As shown in the experiments section, Basic MailRank performs very well in spam detection, while being highly resistant against spammer attacks. However, it still has the limitation of being too general with respect to user ranking. More specifically, it does not address that:

- Users generally communicate with persons ranked average with respect to the overall rankings.
- Users prefer having their acquaintances ranked higher than unknown users, even if one unknown user can achieve a high overall reputation from the network.
- There should be a clear difference between a user’s communication partners, i.e., the ones with a higher rank should be easily recognizable.

⁷Both values, the ‘20%’ and the ‘0.25%’ have been determined in extensive simulations that are not shown here.

⁸To allow new, unknown users to participate in MailRank, an automatically generated email could be sent to the unknown user encouraging her to join MailRank (challenge-response scheme), thus bringing her into the non-spammer area of reputation scores. Similarly, it is necessary to include votes for send-only email addresses, as used very often for order confirmations in eBusiness, to make such send-only addresses distinguishable from spammer addresses.

Algorithm 3.1. The Basic MailRank Algorithm.

Client Side:

Each vote sent to the MailRank server comprises:

$Addr(u)$: The hashed version of the email address of the voter u .

$TrustVotes(u)$: Hashed version of all email addresses u votes for (i.e., she has sent an email to)

Server Side:

1: Combine all received data into a global email network graph. Let T be the Markov chain transition probability matrix, computed as:

ForEach known email address i

If i is a registered address, i.e., user i has submitted her votes

ForEach trust vote from i to j

$T_{ji} = 1/\text{NumOfVotes}(i)$

Else ForEach known address j

$T_{ji} = 1/N$, where N is the number of known addresses.

3: Determine the biasing set B (i.e., the most popular email addr.)

 3a: Manual selection or

 3b: Automatic selection or

 3c: Semi-automatic selection

4: Let $T' = c \cdot T + (1 - c) \cdot E$, with $c = 0.85$ and

$E[i] = [\frac{1}{|B|}]_{N \times 1}$, if $i \in B$, or $E[i] = [0]_{N \times 1}$, otherwise

5: Initialize the vector of scores $\vec{x} = [1/N]_{N \times 1}$, and the error $\delta = \infty$

6: **While** $\delta < \epsilon$, ϵ being the precision threshold

$\vec{x}' = T' \cdot \vec{x}$

$\delta = \|\vec{x}' - \vec{x}\|$

7: Output \vec{x}' , the global MailRank vector.

8: Classify each email address in the MailRank network into: 'spammer' / 'non-spammer' based on the threshold T

Personalizing on each user's acquaintances tackles these aspects. Its main effect is boosting the weight of the user's votes, while decreasing this influence for all the other votes. Thus, the direct communication partners will achieve much higher ranks, even though initially they were not among the highest ones. Moreover, due to the rank propagation, their votes will have a high influence as well.

Now that we have captured the user requirements mentioned, we should also focus our attention on a final design issue of our system: scalability. Simply biasing MailRank on user's acquaintances will not scale well, because it must be computed for each preference set, that is for every registered user.

Jeh and Widom [11] have proposed an approach to calculate Personalized PageRank vectors, which can also be adapted to our scenario, and which can be used with millions of subscribers. To achieve scalability, the resulting personalized vectors are divided in two parts: one common to all users, precomputed and stored offline (called "partial vectors"), and one which captures the specifics of each preference set, generated at run-time (called "hubs skeleton"). We will have to define a restricted set of users on which rankings can be biased (we shall call this set "hub set", and note it with H). There is one partial vector and one hub skeleton for each user from H . Once an additional regular user registers, her personalized ranking vector will be generated by reading the already precomputed partial vectors corresponding to her preference set (step 1), by calculating their hubs skeleton (step 2), and finally by tying these two parts together (step 3). Both the algorithm from step 1 (called "Selective Expansion") and the one from step 2 (named "Repeated Squaring") can be mathematically reduced to biased PageRank. The latter decreases the computation error much faster along the iterations and is thus more efficient, but works only with the output of the former one as input. In the final phase, the two sub-vectors resulted from the previous steps are combined into a global one. The algorithm is depicted in the following lines. To make it clearer, we have also collected the most important definitions it relies on in table 1.

Finally, it should be noted that the original algorithm has been proven to be equivalent to a biased PageRank [11]. Thus, it preserves all the useful properties of the PageRank algorithm (e.g.,

Term	Description
Set V	The set of all users.
Hub Set H	All users the rankings could be personalized on, $H \subset V$.
Preference Set P	Selected set of users to personalize on, $P \subset H$.
Preference Vector p	Preference set with weights.
Personalized PageRank Vector (PPV)	Importance distribution induced by a preference vector.
Basis Vector r_u	PPV for a preference vector with a single nonzero entry at u .
Hub Vector r_u	Basis vector for a hub user $u \in H$.
Partial Vector $r_u - r_u^H$	Used with the hubs skeleton to construct a hub vector.
Hubs Skeleton $r_u(H)$	Used with partial vectors to construct a hub vector.

Table 1: Terms specific to Personalized MailRank.

convergence in the presence of loops in the voting graph, resistance against malicious attacks, etc.), while being much more scalable.

Algorithm 3.2. Personalized MailRank.

0: (Initializations) Let u be a user from H , for which we compute the partial vector, and the hubs skeleton. Also, let $D[u]$ be the approximation of the basis vector corresponding to user u , and $E[u]$ the error of its computation. Initialize $D_0[u]$ with:

$$D_0[u](q) = \begin{cases} c = 0.15 & , q \in H \\ 0 & , otherwise \end{cases}$$

Initialize $E_0[u]$ with:

$$E_0[u](q) = \begin{cases} 1 & , q \in H \\ 0 & , otherwise \end{cases}$$

1: (Selective Expansion) Compute the partial vectors using $Q_0(u) = V$ and $Q_k(u) = V \setminus H$, for $k > 0$, in the formulas below:

$$\mathbf{D}_{k+1}[\mathbf{u}] = \mathbf{D}_k[\mathbf{u}] + \sum_{q \in Q_k(u)} c \cdot E_k[u](q) \mathbf{x}_q$$

$$\mathbf{E}_{k+1}[\mathbf{u}] = \mathbf{E}_k[\mathbf{u}] - \sum_{q \in Q_k(u)} E_k[u](q) \mathbf{x}_q + \sum_{q \in Q_k(u)} \frac{1-c}{|Q_k(u)|} \sum_{i=1}^{|Q_k(u)|} E_k[u](q) \mathbf{x}_{O_i(q)}$$

Under this choice, $D_k[u] + c * E_k[u]$ will converge to $\mathbf{r}_u - \mathbf{r}_u^H$, the partial vector corresponding to u .

2: (Repeated squaring) Having the results from the first step as input, one can now compute the hubs skeleton ($r_u(H)$). This is represented by the final $D[u]$ vectors calculated using $Q_k(u) = H$ into:

$$\mathbf{D}_{2k}[\mathbf{u}] = \mathbf{D}_k[\mathbf{u}] + \sum_{q \in Q_k(u)} E_k[u](q) * D_k[q]$$

$$\mathbf{E}_{2k}[\mathbf{u}] = \mathbf{E}_k[\mathbf{u}] - \sum_{q \in Q_k(u)} E_k[u](q) \mathbf{x}_q + \sum_{q \in Q_k(u)} E_k[u](q) E_k[q]$$

As this step refers to hub-users only, the computation of $\mathbf{D}_{2k}[\mathbf{u}]$ and $\mathbf{E}_{2k}[\mathbf{u}]$ should consider *only* the components regarding users from H , as it significantly decreases the computation time.

3: Let $p = \alpha_1 u_1 + \dots + \alpha_z u_z$ be a preferred vector, where u_i are from H and i is between 1 and z , and let:

$$r_p(h) = \sum_{i=1}^z \alpha_i (r_{u_i}(h) - c * x_{p_i}(h)), \quad h \in H$$

which can be computed from the hubs skeleton.

The PPV v for p can then be constructed as:

$$v = \sum_{i=1}^z \alpha_i (r_{u_i} - r_{u_i}^H) + \frac{1}{c} \sum_{h \in H} r_p(h) * [(r_u - r_u^H) - c * x_h]$$

3.4 MailRank System Architecture

MailRank is composed of a server, which collects all user votes and delivers a score for any known email address, and an email proxy on the client side, which interacts with the MailRank server.

The MailRank Server collects the input data (i.e., the votes) from all MailRank users to run the MailRank algorithm. The votes are assigned with a lifetime for (1) Identifying and deleting email addresses which haven't been used for a long time, and (2) Detecting spammers which behave good for some time to get a high rank and start to send spam emails afterwards.

The MailRank Proxy resides between user's email client and her regular local email server. It performs two tasks: When receiving an outgoing email, it first extracts the user's votes from the available input data (e.g., by listening to ongoing email activities or by analyzing existing sent-mail folders). Then, it sends the votes

to the MailRank server and forwards the email to the local email server. To increase efficiency, only those votes that have not been submitted yet (or that would expire otherwise) are sent. Also, for privacy reasons, votes are encoded using hashed versions of email addresses. Upon receiving an email, the proxy queries the MailRank server about the ranking of the sender address (if not cached locally) and classifies / ranks the email accordingly.

Further extensions of our prototype will make use of secure signing schemes to enable us to analyze both outgoing and incoming emails for extracting the ‘votes’ and submitting them to the MailRank server.⁹ This helps not only to bootstrap the system initially, but also introduces the votes of spammers into MailRank. Such votes have a very positive aspect, since they increase the score for the spam recipients (i.e., non-spammers). Thus, spammers face more difficulties to attack the system and increase their own rank.

3.5 MailRank Under Spammer Attacks

By definition, spammers send the same / very similar message to very many (typically millions of) recipients. However, they can run two different strategies to choose the sender address: First, they use a new (random) email address for each spam message even if they send the same message to millions of recipients (from an analysis we performed on the autowhitelists of several large university institutions in Germany, we found that 95% of the spammer addresses were used only once). In this manner, they are trying to circumvent blacklists of email addresses. Furthermore, they use these addresses only for sending spam emails to non-spammers. Second, they use email addresses from well-known non-spammers (forging of sender address) assuming that these addresses are in the whitelists of many spam detection tools. Sender authentication schemes as those listed in Sect. 2 already prevent forging the sender address (when installed on the email server) and are actually required for any whitelist-based scheme. However, sender authentication cannot counteract the much more common former spamming strategy.

As soon as the MailRank service becomes widespread, spammers will surely try to attack it in order to increase the rank of their own address(es). We identified and simulated several ways of attacking MailRank¹⁰. For example, spammers could issue votes from one or several spammer addresses to one or several non-spammer addresses. However, the algorithm ensures that it is not possible to change your own score by the votes you are issuing towards others. Therefore, such attacks are only reasonable if the spammers vote for another spammer address to increase its rank, forming a *malicious collective* (cf. Fig. 2). This is comparable to link farming in the Web in order to attack PageRank. However, recently there has been an extensive amount of work on identifying and neutralizing such attacks on power iteration algorithms (see for example [20]), and thus the threat they represent to social reputation schemes has been significantly reduced.

Another possible attack is to make non-spammers vote for spammers. To counter incidental votes for spammers (e.g., because of a misconfigured vacation daemon), an additional confirmation process could be required if a vote for one particular email address would move that address from ‘spammer’ to ‘non-spammer’. However, spammers could still *pay* non-spammers to send spam on their

⁹Analyzing incoming votes raises more security issues since we need to ensure that the sender did indeed vote for the recipient, i.e., the vote / email is not faked. This can be achieved by relying on / extending current sender authentication solutions.

¹⁰We refer the reader to [3, 12] for a discussion about attacks in other environments, such as P2P networks, which were also useful as a starting point for analyzing attacks in the MailRank scheme.

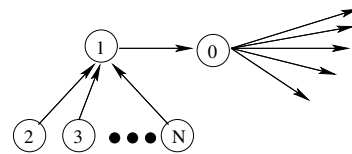


Figure 2: Malicious collective: nodes 2–N vote for node 1 to increase the rank of node 1 and node 1 itself votes for node 0, the email address that is finally used for sending spam emails.

behalf. Such an attack can be successful initially, but the rank of the non-spammer addresses will decrease after some time to those of spammers due to the limited life time of votes. We will discuss simulations based on such attack scenarios in the next section.

4. EXPERIMENTAL RESULTS

Real-world data about email networks is almost unavailable because of privacy reasons. Yet some small studies do exist, using data gathered from the log files of a student email server [4], or of a company wide server [9], etc. In all cases, the analyzed email network graph exhibits a power-law distribution of in-going (exponent 1.49) and out-going (exponent 1.81) links.

To be able to vary certain parameters such as the number of spammers, we evaluated MailRank using an extensive set of simulations, based on a power-law model of an email network, following the characteristics presented in the above mentioned literature studies. Additionally, we used an exponential cut-off at both tails to ensure that a node has at least five and at most 1500 links to other nodes, which reflects the nature of true social contacts [9]. If not noted otherwise, the graph consisted of 100,000 non-spammers¹¹ and the threshold T was set to 0. In a scenario without virus infections, this is sufficient to detect spammers and to ensure that non-spammers are not falsely classified. Furthermore, we repeated all simulations for at least three times with different randomly generated email networks to determine average values. Finally, as personalization brought a significant improvement only in creating user-specific rankings of email addresses (i.e., it resulted only in minor improvements for spam detection), we omitted it here due to space limitations. Therefore, our analysis is focused around three issues: Effectiveness in case of very sparse MailRank networks (i.e., only few nodes submit votes, the others only receive votes), exploitation of spam characteristics, and attacks on MailRank.

4.1 Very Sparse MailRank Networks

In sparse MailRank networks, a certain amount of email addresses only receive votes, but do not provide any because their owners do not participate in MailRank. In this case, some non-spammers in the graph could be regarded as spammers, since they achieve a very low score.

To simulate sparse MailRank networks, we created a full graph as described above and subsequently deleted votes of a certain set of email addresses. We used several removal models:

- All: Votes can be deleted from all nodes.
- Bottom99.9%: Nodes from the top 0.1% are protected from vote deletion.
- Avg: Nodes having more than the average number of outgoing links are protected from vote deletion.

The first model is rather theoretical, as we expect the highly-con-

¹¹We also simulated using 10,000 and 1,000,000 non-spammers and obtained very similar results.

nected non-spammers to register with the system first¹². Therefore, we protected the votes of the top nodes in the other two methods from being deleted¹³. Figure 3 depicts the percentage of non-spammers regarded as spammers, depending on the percentage of nodes with deleted votes, with the error bars at each point showing the minimum / maximum over five simulation runs.

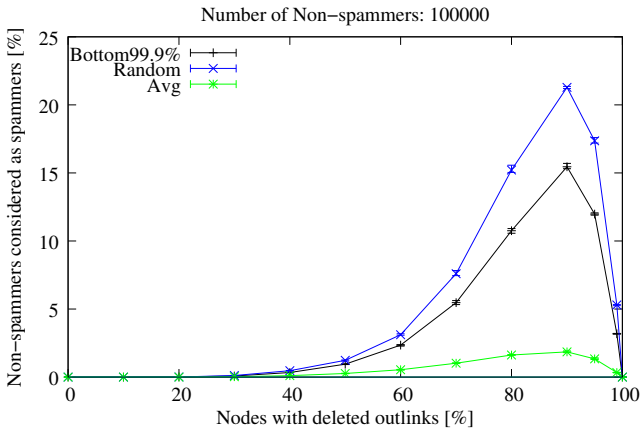


Figure 3: Very sparse MailRank networks

Non-spammers registered to the MailRank system will be classified as spammers only when very few, non-reputable MailRank users send them emails. As studies have shown that people usually exchange emails with at least five partners, such a scenario is rather theoretical. However, as the power-law distribution of email communication is expected only after the system has run for a while, we intentionally allowed such temporary anomalies in the graph. Even though for high deletion rates (70 – 90%) they resulted in some non-spammers being classified as spammers, MailRank still performed well, especially in the more realistic ‘avg’ scenario (the bigger error observed in the theoretical ‘Random’ scenario was expected, since random removal may result in the deletion of high-rank nodes contributing many links to the social network). Finally, the error rate decreases fast when the removal approaches 100%, as the number of nodes known to the system also decreases¹⁴.

4.2 Exploitation of Spam Characteristics

If we monitor current spammer activities (i.e., sending emails to non-spammers), the emails / votes from spammers towards non-spammers can be introduced into the system as well. This way, spammers actually contribute to improve the spam detection capabilities of MailRank: The more new spammer email addresses and emails are introduced into the MailRank network, the higher they increase the score of the receiving non-spammers. This can be seen in a set of simulations with 20,000 non-spammer addresses and a varying number of spammers (up to 100,000, cf. Fig. 4), where the rank of the top 0.25% non-spammers increases linearly with the number of spammer addresses included in the MailRank graph.

4.3 Attacking MailRank

In order to be able to attack MailRank, spammers must receive votes from other MailRank users to increase their rank. As long

¹²Such behavior was also observed in real-life systems, e.g., in the Gnutella P2P network (<http://www.gnutella.com/>).

¹³The 100% from ‘Bottom99.9%’ and ‘avg’ actually refer to 100% of the non-protected nodes.

¹⁴When all users pointing to a not registered user have been deleted, then the not registered user is no longer known to the system.

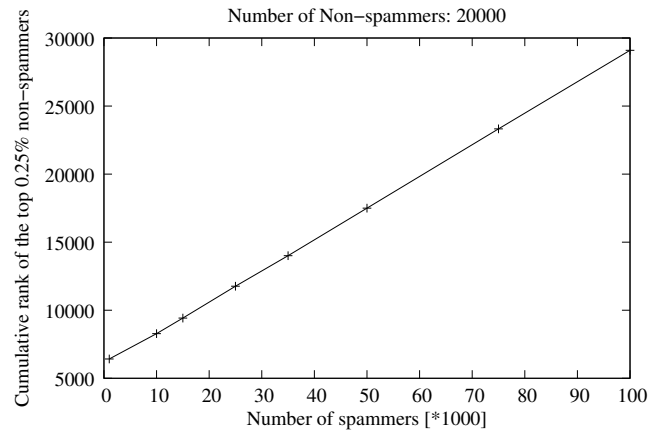


Figure 4: Rank increase of non-spammer addresses

as nobody votes for spammers, they will achieve a null score and will thus be easily detected. This leaves only two ways of attacks: formation of malicious collectives and virus infections.

Malicious collectives. The goal of a malicious collective (cf. Fig. 2) is to aggregate enough score into one node to push it into the biasing set. If no manually selected biasing set can be used to prevent this, one of the already many techniques to identify web link farms could be employed (see for example [20]). Furthermore, we require MailRank users willing to submit their votes to manually register their email address(es). This impedes spammers to automatically register millions of email addresses in MailRank and also increases the cost of forming a malicious collective. To actually determine the cost of such a manual registration, we have simulated a set of malicious users as shown in Fig. 2. The resulting position of node 1, the node that should be pushed into the biasing set, is depicted in Fig. 5 for an email network of 20,000 non-spammers, malicious collectives of 1000 nodes each, and an increasing number of collectives on the x-axis. When there are few large-scale

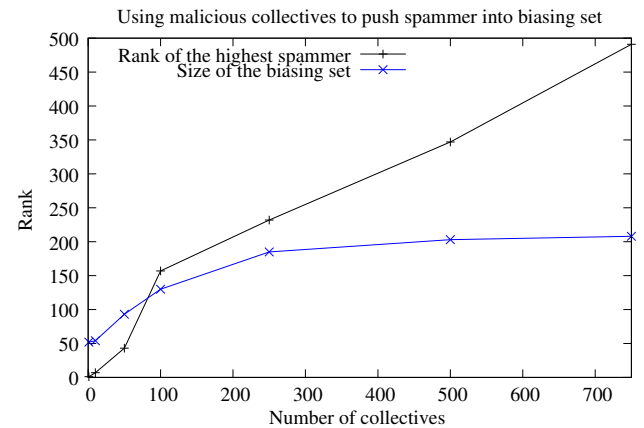


Figure 5: Automatic creation of the biasing set

spammer collectives, the system could be relatively easy attacked. However, as users must manually register to the system, forming a collective of sufficient size is practically infeasible. Moreover, in a real scenario there will be more than one malicious collective, in which case pushing a node into the biasing set is almost impossible: As shown in Fig. 5, it becomes more difficult for a malicious

collective to push one node into the biasing set, the more collectives exist in the network. This is because the spammers registered to the system implicitly vote for the non-spammers upon sending them (spam) emails.

Virus infections. Another possible attack on MailRank is to use virus / worm technology to infect non-spammers and make them vote for spammers. We simulated such an attack according to Newman's studies [14], which showed that when the 10% most connected members of a social network are not immunized (e.g., using anti-virus applications) worms would spread too fast. Simulation results are shown in Fig. 6 with a varying amount of non-spammers voting for 50% of all spammers. If up to about 25%

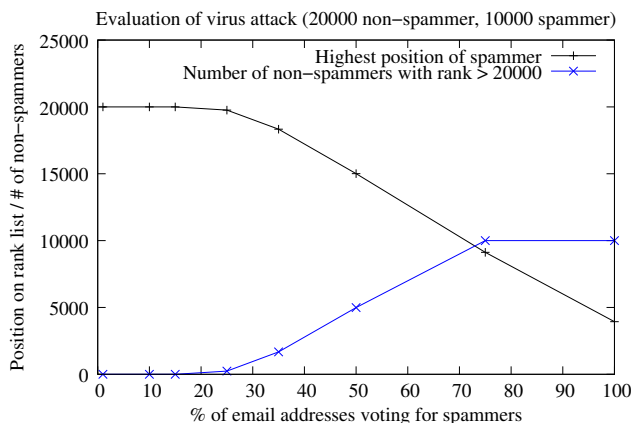


Figure 6: Simulation results: Virus attack

of the non-spammers are infected and vote for spammers, there is still a significant difference between the ranks of non-spammers and spammers, and no spammer manages to get a higher rank than the non-spammers. If more than 25% non-spammers are infected, the spammer with the highest rank starts to move up in the rank list (the upper line from Fig. 6 descends towards rank 1). Along with this, there will be no clear separation between spammers and non-spammers, and two threshold values must be employed: one MailRank score T_1 above which all users are considered non-spammers and another one $T_2 < T_1$ beneath which all are considered spammers, the members having a score within (T_1, T_2) being classified as unknown.

5. CONCLUSIONS AND FUTURE WORK

This paper investigated the feasibility of MailRank, a new email ranking and classification scheme, which intelligently exploits the social communication network created via email interactions. On the resulting email network graph, a power-iteration algorithm is used to rank trustworthy senders and to detect spammers. MailRank performs well both in the presence of very sparse networks: Even in case of a low participation rate, it can effectively distinguish between spammer email addresses and non-spammer ones, even for those users not participating actively. MailRank is also very resistant against spammer attacks and, in fact, has the property that when more spammer email addresses are introduced into the system, the performance of MailRank increases.

Based on these encouraging results we are currently investigating several future improvements for our algorithms. We intend to move from a centralized system to a distributed one to make the system scalable for a large-scale deployment. We are currently investigating a DNS-like system, in which the computation is handled in a distributed manner by several servers. Finally, another

approach would be to consider each email client as a peer in a P2P network, and run a distributed approach to MailRank as such.

6. REFERENCES

- [1] P.O. Boykin and V. Roychowdhury. Leveraging social networks to fight spam. *IEEE Computer*, 38(4):61–68, 2005.
- [2] Paul-Alexandru Chirita, Daniel Olmedilla, and Wolfgang Nejdl. Finding related pages on the link structure of the www. In *Proc. of the 3rd IEEE/WIC/ACM International Web Intelligence Conference*, Sep 2004.
- [3] A. Clausen. The Cost of Attack of PageRank. In *Proc. of the International Conference on Agents, Web Technologies and Internet Commerce (IAWTIC)*, Gold Coast, 2004.
- [4] H. Ebel, L. I. Mielsch, and S. Bornholdt. Scale-free topology of email networks. *Physical Review E* 66, 2002.
- [5] D. Geer. Will new standards help curb spam? *IEEE Computer*, pages 14–16, February 2004.
- [6] J. Golbeck and J. Hendler. Reputation Network Analysis for Email Filtering. In *Proc. of the Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, July 2004.
- [7] A. Gray and M. Haahr. Personalised, Collaborative Spam Filtering. In *Proc. of the Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, July 2004.
- [8] Z. Gyöngyi, H. Garcia-Molina, and J. Pendersen. Combating web spam with trustrank. In *Proc. of the 30th International VLDB Conference*, 2004.
- [9] B. A. Huberman and L. A. Adamic. Information dynamics in the networked world. *Complex Networks, Lecture Notes in Physics*, 2003.
- [10] Isode. Benchmark and comparison of spamassassin and m-switch anti-spam. Technical report, Isode, April 2004.
- [11] G. Jeh and J. Widom. Scaling personalized web search. In *Proc. of the 12th Intl. WWW Conference*, 2003.
- [12] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *Proc. of the 12th Intl. WWW Conference*, 2003.
- [13] J.S. Kong, P.O. Boykin, B.A. Rezaei, N. Sarshar, and V. Roychowdhury. Let your CyberAlter Ego Share Information and Manage Spam. Technical report, University of California, USA, 2005. Preprint.
- [14] M. E. J. Newman, S. Forrest, and J. Balthrop. Email networks and the spread of computer viruses. *Physical Review E* 66, 2002.
- [15] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.
- [16] M. Perone. An overview of spam blocking techniques. Technical report, Barracuda Networks, 2004.
- [17] P. Resnick and H.R. Varian. Recommender Systems. *Communications of the ACM*, 40(3):56–58, 1997.
- [18] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proc. of the 2nd International Semantic Web Conference*, 2003.
- [19] G.L. Wittel and S.F. Wu. On Attacking Statistical Spam Filters. In *Proc. of the Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, July 2004.
- [20] B. Wu and B. Davison. Identifying link farm spam pages. In *Proc. of the 14th Intl. WWW Conference*. ACM Press, 2005.
- [21] C. Ziegler and G. Lausen. Spreading activation models for trust propagation. In *Proc. of the IEEE Intl. Conference on e-Technology, e-Commerce, and e-Service*, 2004.