Leveraging Social Networks to Fight Spam

Social networks are useful for judging the trustworthiness of outsiders. An automated antispam tool exploits the properties of social networks to distinguish between unsolicited commercial e-mail—spam—and messages associated with people the user knows.

P. Oscar Boykin
University of Florida

Vwani P. Roychowdhury
University of California, Los Angeles

The amount of unsolicited commercial e-mail—spam—and, more importantly, the portion of e-mail that is spam, has increased dramatically in the past few years. A recent study showed that 52 percent of e-mail users say spam has made them less trusting of e-mail, and 25 percent say that the volume of spam has reduced their e-mail use.[1]

This crisis has prompted proposals for a broad spectrum of potential solutions, ranging from more efficient antispam software tools to antispam laws at both the federal and state levels.

While the jury is still out on how widely such antispam laws will be enacted and how effectively they would stem the flow, the objective of the various legal and technical solutions is the same: to make sending spam unprofitable and thereby destroy the spammers' underlying business model.

Achieving these goals requires widespread deployment and use of antispam techniques. To gain user confidence, which is a prerequisite for wide deployment, the tool must be accurate, user friendly, and computationally efficient.

Our technique simultaneously achieves all three requirements. This technique is predicated on recognizing the unique characteristics inherent to social networks and the proven wisdom of using such trust networks to make the right choices. The reliability of our decisions, then, depends strongly on the trustworthiness of our underlying social networks. Thus, we seem to have evolutionarily developed several interaction strategies that can generate a trustworthy network.

A commonly espoused rule suggests that trust is built based not only on how well you know a person but also on how well others in your network know the person. This interaction dynamic results in close-knit communities in which, if Alice knows Bob and Charlotte, it's highly likely that Bob and Charlotte also know each other.

We show that this natural instinct to form close-knit social networks operates in cyberspace as well and can be exploited to provide an effective and automated spam-filtering algorithm.

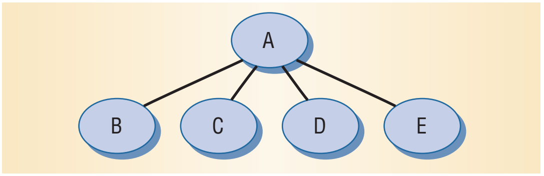

Figure 1. Star subnetwork. Subgraph represents e-mail interaction generated by a message sent from A to B and C and cc'd to D and E.

PERSONAL E-MAIL NETWORKS

Some researchers have constructed e-mail graphs based on e-mail address books[2] or complete e-mail logs of sets of users.[3-5] We based our network on the information available to just one user of an e-mail system—specifically, the headers of all of the e-mail messages in that user's inbox. Each e-mail header contains the sender's e-mail address, stored in the "From" field, and a list of recipient addresses, stored in the "To" and "Cc" fields.

To construct a personal e-mail network, we first created nodes representing all of the addresses appearing in all of the e-mail headers of the messages in the user's inbox. We added edges between pairs of addresses appearing in the same header—that is, addresses of individuals who have communicated via the user. For example, suppose Alice sends a message "To" Bob and Charlotte, with a "Cc" to David and Eve. We represented this e-mail interaction using a star subnetwork, as Figure 1 illustrates.

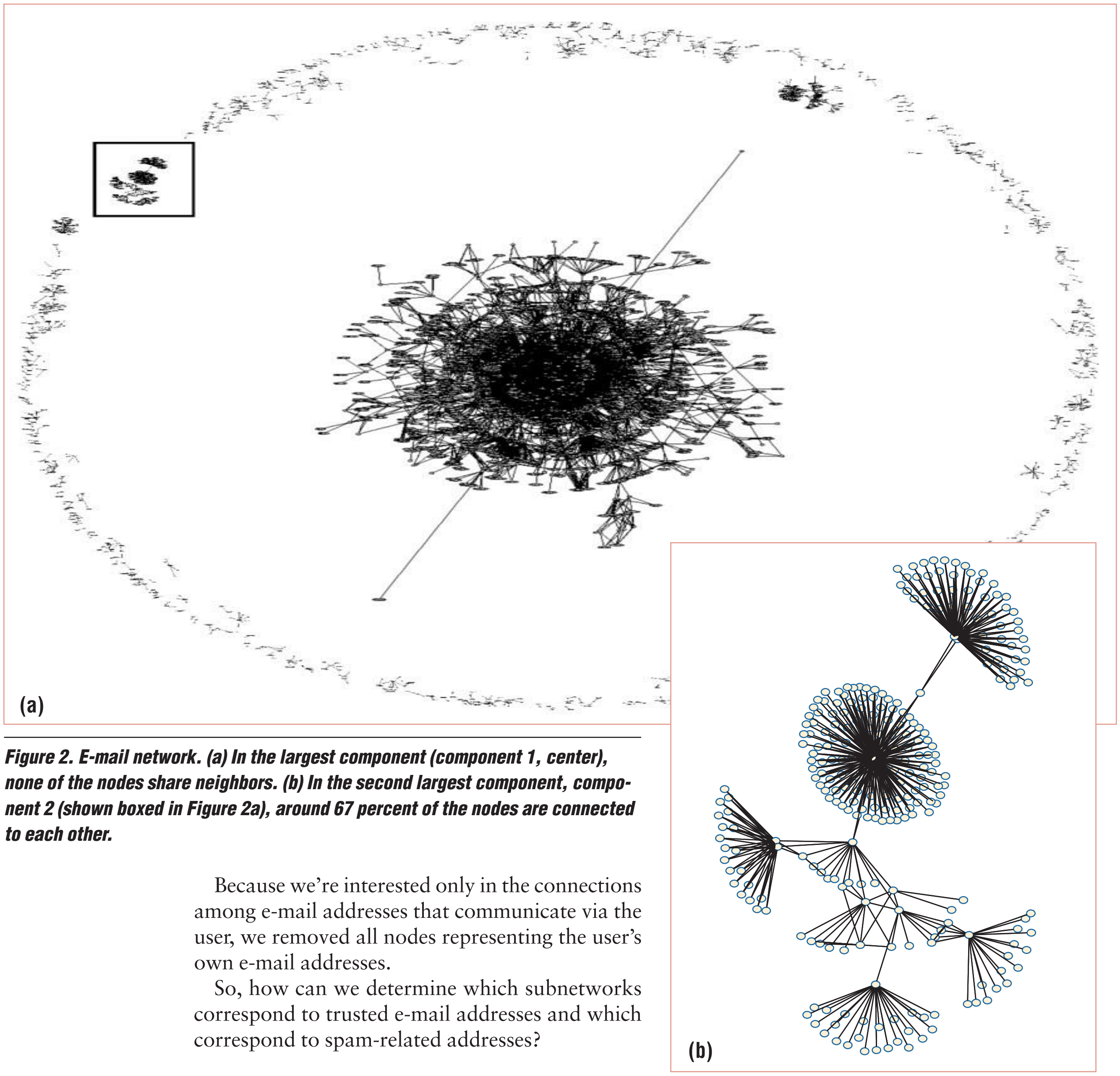

Figure 2. E-mail network. (a) In the largest component (component 1, center), none of the nodes share neighbors. (b) In the second largest component, component 2 (shown boxed in Figure 2a), around 67 percent of the nodes are connected to each other.

Because we're interested only in the connections among e-mail addresses that communicate via the user, we removed all nodes representing the user's own e-mail addresses.

So, how can we determine which subnetworks correspond to trusted e-mail addresses and which correspond to spam-related addresses?

SOCIAL NETWORKS AND WHITE LISTS

Many recent studies have identified quantitative measures of a community's closeness and have used these measures to distinguish empirically observed social networks from less-known, nonsocial networks.[4,6,7]

A social network's most distinctive property is the tendency to cluster. For example, if Alice knows Bob and Eve in a social network, Bob is considerably more likely to know Eve than in a random network with similar degree distribution.

To define a qualitative expression for a network's clustering coefficient, we begin by counting all pairs of nodes that are part of a wedge—that is, each node in the pair has a direct edge to a third node.

According to the intuitive notion of clustering, in a graph with a high clustering coefficient, many of these pairs will also be connected by an edge—that is, many of the wedges also form triangles. Hence, we can express the clustering coefficient—sometimes called transitivity—C of a graph as:

$$C = \frac{3 \times \text{(number of triangles in the graph)}}{\text{number of wedges}} \quad (1)$$

This expression should provide an idea of the clustering coefficient's physical meaning in social networks.

We use a quantitative definition of the clustering coefficient that involves counting the fraction of a node's neighbors that are also each other's neighbors.

Specifically, a node with degree k_i has k_i neighbors. Each k_i neighbor is potentially connected to the others. A total of $k_i(k_i - 1) = 2$ possible connections exists between the neighbors. Counting the number of connections E_i and dividing by $k_i(k_i - 1) = 2$ gives us the node's clustering coefficient.

Because the quantity for nodes of degree 1 is undefined, we only count nodes of degree greater than 1. The clustering coefficient for the entire graph is the average of the clustering coefficient for each node (of degree greater than 1)

$$C = \frac{1}{N_2} \sum_i \frac{2E_i}{k_i\,(k_i - 1)} \qquad (2)$$

where N_2 is the total number of network nodes of degree 2 or greater.

Other researchers have applied this metric to e-mail graphs and found that the clustering coefficient is more than 10 times larger than we would expect from a random graph with the same degree distribution.[4]

To demonstrate how to use a clustering coefficient to distinguish between spam and nonspam e-mail, consider the connected components 1 and 2 in the personal e-mail network consisting of 5,486 messages that is depicted in Figure 2. Interestingly, and perhaps contrary to intuition, the largest connected component in this particular network corresponds to spam-related e-mail.

Component 1 (Figure 2a) has a clustering coefficient of 0: Exactly zero nodes share neighbors with any of their neighbors. On the other hand, component 2 (Figure 2b), which is smaller in size, has a clustering coefficient of 0.67: Around 67 percent of each node's neighbors are connected to each other.

Figure 3, a subgraph of component 1, and Figure 4, a subgraph of component 2, show the relative incidence of the triangle structures that characterize close-knit communities.

Given that social networks have high clustering coefficients, we can be confident that the e-mail addresses in component 2 are part of the user's cyberspace social network. Thus, we can classify any e-mail with nodes from the second component in its header as nonspam. The e-mail addresses associated with these nodes comprise the user's *white list*.

BLACK LISTS AND SPAM COMPONENT FORMATION

If we can likewise conclude that spam generates the first component, which has a low clustering coefficient, we can label any e-mail sent or coreceived by a node inside the first component as spam.

Indeed, a detailed bookkeeping of the inbox shows that the e-mail addresses in component 1 are

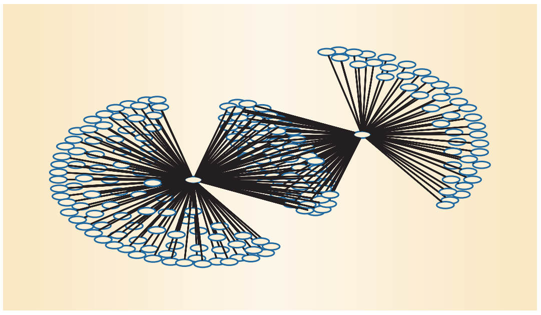

Figure 3. Subgraph of a spam component. (a) Two spammers share many corecipients (middle nodes). In this subgraph, no node shares a neighbor with any of its neighbors.

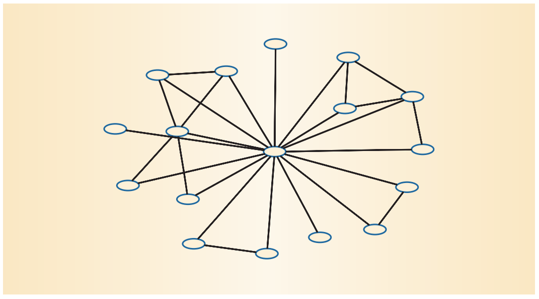

Figure 4. Subgraph of a nonspam component. The nonspam graph shows a higher incidence of triangle structures (neighbors sharing neighbors) than the spam subgraph.

always related to spam, just as those in component 2 are always part of the user's social network. In other words, the e-mail addresses in the first component comprise a *black list*.

Although we can expect networks of friends to have a high clustering coefficient, it is important to understand why a large subnetwork with a low clustering coefficient is likely to be spam-induced.

A careful analysis of component 1 reveals that it was created by a *dictionary attack*, a common spamming technique in which spammers send messages to corecipients sorted by e-mail address. For example, adam@example.com will likely have corecipients with alphabetically similar e-mail addresses, such as arthur@example.com, alex@example.com, and avid@example.com. Because of sorted recipient lists, corecipients are often repeated in different spam e-mail messages, causing the disconnected spam components to merge into larger components.

The spam messages form a bipartite graph, with two types of nodes representing spammers and spam recipients. Because the spammers don't spam each other, and the corecipients of spam messages don't know each other, this graph will always have a clustering coefficient of 0.

To determine how quickly the spammer components will merge, we can examine the probability that two different spammers send messages to the same corecipient.

Figure 5 shows the complementary cumulative distribution function (CCDF) of the number of corecipients per spam message. In this data, the average number of recipients of a spam message is 3.87. As a model, we assume that each spammer uses a "From" address only once and sends the spam to l recipients, which the spammer chooses at random. Based on our data, we choose $l \approx 3.87$.

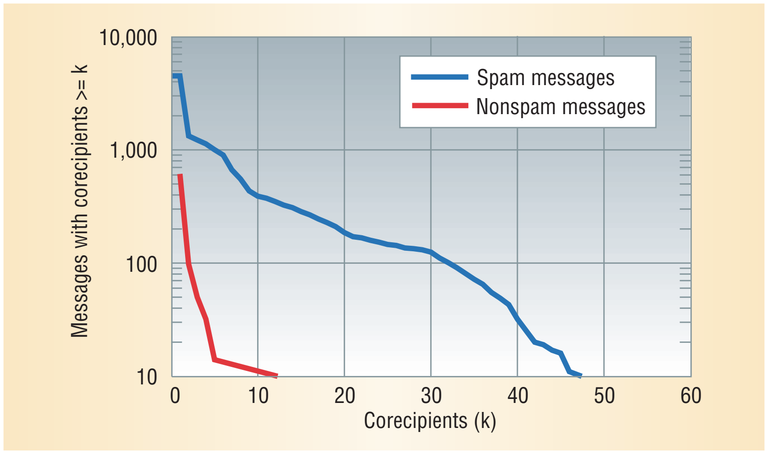

***Figure 5.* Complementary cumulative distribution function (CCDF) of the number of corecipients per message for spam and nonspam messages. The *x*-axis shows the number of corecipients, and the *y*-axis shows the number of messages with at least that number of corecipients. The mean number of corecipients for the spam messages is 3.87; for nonspam, the number is 1.71.**

The model assumes that k e-mail addresses are near the user's address in the sorted address space.

To solve for the size of the largest connected component, we define S_i as the size of the largest connected component after i messages. The probability that each recipient is already in the largest component is $S_i = k/q$. The probability that m recipients are in the largest component is

$$p_m = \binom{l}{l-m}(1-q)^{l-m}q^m$$

Combining this in a rate equation, we find

$$\begin{aligned} S_{i+1} &= \sum_{m=0}^{l} p_m\,(S_i + (l-m)) - p_0 l \\ &= S_i + (1-p_0)l - ql \\ &= S_i - \frac{l}{k}S_i + (1-(1-q)^l)l \end{aligned}$$

$$\frac{dS_i}{di} \approx (1-(1-q)^l)l - \frac{l}{k}S_i$$

We approximate this equation in two regimes—unsaturated ($q << 1$) and saturated ($q \approx 1$)—to get two different solutions. In the unsaturated regime,

$$\begin{aligned} \frac{dS_i}{di} &\approx (1-(1-q)^l)l - \frac{l}{k}S_i \\ &\approx l^2q - \frac{l}{k}S_i \\ &= \frac{l(l-1)}{k}S_i \\ \Rightarrow S_i &= le^{\frac{l(l-1)}{k}i} \end{aligned}$$

In the saturated regime,

$$\begin{aligned} \frac{dS_i}{di} &\approx (1-(1-q)^l)l - \frac{l}{k}S_i \\ &\approx l - \frac{l}{k}S_i \\ \Rightarrow S_i &= k(1-e^{-\frac{1}{k}i}) + le^{-\frac{1}{k}i} \end{aligned}$$

Clearly, after $O(k/l^2)$ messages, the spam components should start to join. This analysis underestimates the joining rate because it assumes that a message only joins with the largest component and never adds more than one component, which clearly only increases the rate that the spam components grow in size.

We've also ignored the fact that the nearer the addresses are by alphabetical measure, the more likely they are to be corecipients of an e-mail message. Instead, we approximate that k nearby addresses exist and that the spammer selects them randomly.

GRAY LISTS AND AMBIGUOUS SUBGRAPHS

We can also use this scheme to classify all components of a user's personal network.

For each component, we note the size, maximum degree (k_{max}), and clustering coefficient. We expect a minimum number of nodes for which we can reliably measure a component's clustering coefficient and exclude components smaller than this cutoff size (S_{min}) from our classification scheme. We also know that graphs with power-law exponents greater than or equal to –2.0 will have a maximum degree on the order of the graph's size.[6] Like previous work,[4,5] we find degree distributions with exponents greater than –2.0.

We use this fact to introduce another cutoff parameter. To limit a single node's impact on the statistics, we disregard all components with a clustering coefficient equal to 0 and (k_{max} + 1)/size greater than K_{frac}. We assume the remaining components with clustering coefficients less than a critical value C_{min} are spam components and write all nodes in these components to the black list.

If a component's clustering coefficient is greater than C_{max}, we write all nodes to the white list. In rare cases, we include a component with a clustering coefficient between C_{min} and C_{max}, as we discuss later.

To choose the cutoff parameters S_{min} and K_{frac}, we used several criteria. Single messages can make isolated components, which are difficult to classify a priori, because every message with k corecipients can create a component with a clustering coefficient equal to 0.0 and size k. Setting K_{frac} to less than 1.0 (0.6 to 0.8 works well in practice) ensures that the algorithm won't consider a component from a single message.

A more direct route (but one that can't be realized using purely graph-theoretic methods) is to only consider components formed by N or more messages. The size cutoff should come from a message's expected number of recipients. Setting this far above the mean also ensures that each component has several messages. For the data we examined, a cutoff size of 10 to 20 messages seemed to work well.

Choosing C_{min} and C_{max} involved another set of criteria. Although we expect $C = 0$ for spam components, in our data, as in previous studies,[4,6] the clustering coefficient of the social graphs was an order of magnitude larger than we would expect for a random graph with the same degree distribution. We found that $C_{min} = 0.01$ and $C_{max} = 0.1$ produced excellent results.

Further research on personal e-mail networks will help improve the methods for choosing parameters. This research will provide a better understanding of the statistical properties of these networks over a larger set of users.

Purely by chance, a spammer might send a spam message to a user and one of his or her friends (we assume here that spammers don't know the e-mail addresses of a user's friends). This event will be rare because we imagine there is a very small probability that each corecipient on a spam message is a friend. Hence, spam components usually stay disconnected from friend components.

The possibility exists, however, that a spam message will have a corecipient in common with a nonspam message. The cross-component connections are most likely in spam components with a large number of corecipients. This means that chance connections will result in a nonspam component joined with a large spam component through a small number of edges (the chance corecipient connections).

From a graph-theoretic perspective, this situation corresponds to two connected communities that have few edges between them and different clustering coefficients. Large components with intermediate clustering coefficients typically signal these cases. For example, if the component size is large but the clustering coefficient is less than C_{max}, we can assume a joined spam component.

Edge betweenness is a proposed metric for identifying edges between communities.[3,8,9] The betweenness of an edge within a network is the number of shortest paths between all of the pairs of nodes in the network that include the edge.

All paths linking the nodes in the spam community to nodes in the nonspam community in a joined component will include one of the edges corresponding to the chance connections between the two communities. Therefore, these edges will have a much higher betweenness than the edges connecting members in the same community. Therefore, we split joined components into two communities by removing the edges with the highest betweenness until we have two distinct components.

Newman and Girvan's algorithm[8] uses the same step; however, we don't execute the step on components with a high clustering coefficient. In fact, Newman and Girvan's community-finding algorithm tends to cut these e-mail graphs into many communities, whereas we're only interested in finding the split between spammers and nonspammers.

Although we rarely need the separation technique, it's a robust method for separating joined components, as long as spammers don't have access to the user's white list. For example, in our analysis of a data set containing all of a user's saved e-mail messages over a period of three years, we found only one example in which there was an edge between a spam and nonspam community. We used the separation technique to remove this edge.

In the absence of a complete experimentally verified theory for how personal e-mail networks are formed, we must rely on empirical observations of such networks. Although no theory exists that requires the graph parameters we assume here, some models based on underlying community structure have sought to explain the high clustering coefficient.[6,7] Access to mail headers of many different users might help researchers find a theory to explain the mechanisms giving rise to the properties our algorithm uses.

Presumably, all users are subject to a similar spammer environment—that is, spammers attack all users similarly. As such, until spammers change their tactics, the algorithm's black-listing power should apply to all users. If some users' social structures don't match those previously measured—specifically, if their personal e-mail networks exhibit extremely low clustering—the algorithm might misclassify nonspam messages as spam (false positives).

Table 1. Algorithm results for three data sets. (Data set 1 is from a six-week period; sets 2 and 3 are from five-week periods.)

Data	Black list	White list	Gray list	Total
Spam 1	1,664	0	2,841	4,505
Nonspam 1	0	331	282	613
Spam 2	2,988	0	1,142	4,130
Nonspam 2	0	66	215	281
Spam 3	785	0	297	1,082
Nonspam 3	0	88	461	549

EXPERIMENTAL RESULTS

Table 1 shows the results of using the algorithm on three data sets covering three users over five- and six-week periods. Averaging across users, 34 percent of the nonspam is on the white list, 56 percent of the spam is on the black list, and 47 percent

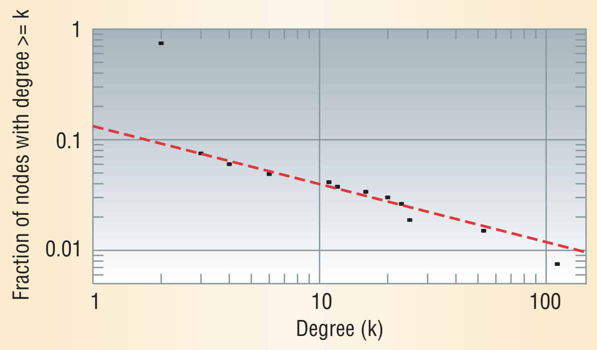

Figure 6. Degree CCDF for the largest nonspam component. The line corresponds to $p_k \propto k^{-1.52}$. Nodes with degrees 1 and 2 don't fit the tail's power law. This occurs because the personal e-mail network construction adds an edge only when the user is a recipient and doesn't see the edges that would exist between two users if it considered both users' e-mail traffic.

of the messages are on the gray list. Thus, the algorithm correctly identifies the message about half the time; in the remaining half, it can't classify the message. Over the test data, it correctly classifies all messages.

To improve our algorithm, we added more graph-theoretic parameters to the classification scheme. Figure 6 shows the CCDF of the largest nonspam component. The degree distribution for the CCDF's tail is $p_k \propto k^{-1.52}$. The nodes with degrees 1 and 2 don't fit the tail's power law. Making a subgraph of the entire e-mail network from a single user's view produces a degree distribution consistent with work based on multiple users' e-mail.[4,5]

Previous studies found degree distributions following power laws with exponents from –1.3 to –1.8. In our data sets, the spam components had power laws with exponents between –1.8 and –2.0.

Thus, because spam components seem to have unusually high power-law exponents, we might be able to use a component's degree distribution to improve the distinguishing power and thus reduce the likelihood of error.

USING THE ALGORITHM

Our e-mail-network-based spam-filtering algorithm automatically constructs white lists with a 34 percent success rate, black lists with a 56 percent success rate with no false classifications, and leaves 47 percent of the e-mail unclassified. It achieves these hit rates without any user intervention.

Table 2. Results of a Bayesian classifier trained on the data from the graph algorithm.

Data	Percent classified as spam	Percent classified as nonspam
Spam 1	80.8	19.2
Nonspam 1	0.3	99.7
Spam 2	97.6	2.4
Nonspam 2	48.8	51.2
Spam 3	95.6	4.4
Nonspam 3	4.7	95.3

Because the only information the algorithm needs is available in the user's e-mail headers, we can easily implement it at various points in the filtering process.

Existing white-list systems use a list of acceptable e-mail addresses or mail servers and assume that all senders not on the list are spam. Similarly, existing black list systems block mail from e-mail addresses or mail servers on a list, and assume all other senders are nonspam. These lists are currently created in an ad hoc manner, based on global prior knowledge and user feedback.

Thus, we can easily integrate our technique, based purely on graph-theoretic methods, with existing methods. For example, administrators of large e-mail servers, such as corporate e-mail servers or Internet service providers, can use our algorithm because they can generate a personal e-mail network for each user as their mail servers receive mail. The ability to generate white lists and black lists for all users will greatly improve the central mail servers' ability to reduce the number of spam messages that reach end users.

The algorithm is also virtually immune to false positives, suggesting that our method can also significantly increase the ease of use of content-based antispam tools, which classify e-mail as spam or nonspam based on content rather than the sender's address.

To illustrate how we integrate our graph-based algorithm with a learning algorithm, we created a simple example using the CRM114 Bayesian classifier.[10]

When performing Bayesian learning,[11,12] it's important not to overtrain and, in the case of e-mail, not to prefer false positives (spam misclassified as nonspam) to false negatives (nonspam misclassified as spam).

We had the classifier learn all of the messages classified as nonspam, and we had it classify each spam message. If the classifier misclassified one of these messages, we set it to learn the message as spam. In other words, we trained the classifier on all of the white list, but only used train-on-error for the black list. It eventually classified all messages as spam or nonspam. Table 2 lists the results.

In this case, data sets 1 and 3 performed rather well, but data set 2 performed quite poorly. We believe this is because data set 2 had the fewest nonspam messages to learn and also the worst ratio of spam to nonspam: 14.7, compared to 7.35 and 1.97 for data sets 1 and 3.

These preliminary results are only estimates of possible performance. Future research should produce more sophisticated training schemes to better combine our algorithm with content-based learning methods.

COUNTERMEASURES

Spammers, of course, will try to defeat antispam tools. A few countermeasures might foil our algorithm. The most obvious countermeasure is to never use multiple recipients in a spam message's "To" or "Cc" headers. This would make the spam components isolated nodes in our graph, which the algorithm would disregard when constructing the black list. However, this wouldn't impact users' ability to automatically generate white lists, which are extremely valuable in improving the accuracy of content-based filtering systems.

Spammers could also attempt to make the algorithm misclassify them as nonspammers. For example, a spammer could use spyware to try to learn e-mail addresses for each user's friends. The spammer could then include the user's friends as corecipients of a spam message, thus posing as a member of the user's social network. If spammers can impersonate members of a white list, however, they can damage the effectiveness of any white-listing scheme, not just our algorithm.

In another countermeasure, spammers construct an artificial social network and attempt to have their messages white-listed. The spammer then sends e-mail messages in which the "From," "To," and "Cc" headers match a real social network's structure. The spammer could include the spam targets—those users for whom the spam is intended—as "Bcc" recipients, which don't show up in the headers. Although our algorithm could label this component as part of the white list, this countermeasure itself could be countered. (Recall that we use only incoming mail headers, not outgoing mail headers—that is, the mail sent by algorithm users.)

By using outgoing mail headers, we can disregard any nonspam components to which the user has never sent any mail. Although this countermeasure can damage the user's ability to construct black lists, it doesn't appear to damage the user's ability to construct white lists. Hence, at this time, we see no countermeasures that could destroy the algorithm's white-listing aspect.

Many areas are open for improving our spam-filtering algorithm. Most obviously, we need to include more component parameters to distinguish spam from nonspam. An algorithm that incorporates more parameters could potentially reduce the probability of misclassification even further.

Unfortunately, significantly decreasing the gray list's size doesn't appear to be a promising option because the gray list components are very small and thus inhabit a small parameter space. Hence, the ability to strongly distinguish between gray-listed components appears to be beyond purely graph-based techniques.

Using cryptographic white lists, in which a user only accepts messages cryptographically signed by authenticated keys, is clearly a solution to the spam problem. Unfortunately, without an infrastructure to make the scheme accessible to most end users, its immediate potential for widespread use is highly questionable. Until this infrastructure is in place, end users must rely on less-perfect solutions, and they will benefit from any effort that makes these solutions more user friendly and easier for mail servers and ISPs to broadly distribute.

As long as spammers continue to adapt their strategies for defeating antispam tools, improving these tools is a subject that will warrant further study. However, even if the spam problem is solved, our e-mail network tool will become increasingly useful. In particular, using a personal e-mail network has the potential to capture the community structure in cyberspace. It's possible that better e-mail message management can be achieved if e-mail clients are aware of the user's various social groups. Our scheme for generating personal e-mail networks could provide such information without any changes to Internet e-mail protocols. ■

Acknowledgments

We thank Jesse Bridgewater for use of his e-mail, Joseph Kong and Kamen Lozev for their comments, and Julie Walters for use of her e-mail and her helpful suggestions. US National Science Foundation grant ITR0326605 partially supported this work.

References

1. D. Fallows, *Spam: How It Is Hurting E-Mail and Degrading Life on the Internet*, tech. report, Pew Internet and American Life Project, Oct. 2003; http://www.pewinternet.org/reports/toc.asp?Report=102.
2. M.E.J. Newman, S. Forrest, and J. Balthrop, "E-Mail Networks and the Spread of Computer Viruses," *Physical Rev. E*, vol. 66, no. 035101, 2002.
3. J.R. Tyler, D.M. Wilkinson, and B.A. Huberman, "E-Mail as Spectroscopy: Automated Discovery of Com-

munity Structure within Organizations," preprint, http://xxx.lanl.gov/abs/cond-mat/0303264.
4. H. Ebel, L.-I. Mielsch, and S. Bornholdt, "Scale-Free Topology of E-Mail Networks," *Physical Rev. E*, vol. 66, no. 035103, 2002.
5. G. Caldarelli, F. Coccetti, and P.D.L. Rios, "Preferential Exchange: Strengthening Connections in Complex Networks," *Physical Rev. E*, vol. 70, no. 027102, 2004.
6. M.E.J. Newman and J. Park, "Why Social Networks Are Different from Other Types of Networks," *Physical Rev. E.*, vol. 68, no. 036122, 2003.
7. M.E.J. Newman, "Assortative Mixing in Networks," *Physical Rev. Letters*, vol. 89, no. 208701, 2002.
8. M.E.J. Newman and M. Girvan, "Finding and Evaluating Community Structure in Networks," *Physical Rev. E*, vol. 69, no. 026114, 2004.
9. M.E.J. Newman, "Fast Algorithm for Detecting Community Structure in Networks," *Physical Rev. E*, vol. 69, no. 066133, 2004.
10. W.S. Yerazunis, "Sparse Binary Polynomial Hashing and the CRM114 Discriminator," presented at the MIT Spam Conf., 2003; http://crm114.sourceforge.net/CRM114_paper.html.
11. M. Sahami et al., "A Bayesian Approach to Filtering Junk E-Mail," *Learning for Text Categorization: Papers from the 1998 Workshop*, AAAI tech. report WS-98-05, AAAI Press, 1998.
12. D. Heckerman, "A Tutorial on Learning with Bayesian Networks," *Learning in Graphical Models*, M. Jordan, ed., MIT Press, 1998.

***P. Oscar Boykin** is an assistant professor of electrical and computer engineering at the University of Florida. His research interests include quantum cryptography, quantum information and computation, peer-to-peer computing, and neural coding. Boykin received a PhD in physics from the University of California, Los Angeles. Contact him at boykin@ece.ufl.edu.*

***Vwani P. Roychowdhury** is a professor of electrical engineering at the University of California, Los Angeles. His research interests include models of computation, including parallel and distributed processing systems, quantum computation, and information processing; adaptive and learning algorithms; nonparametric methods and algorithms for large-scale information processing; and information theory. Roychowdhury received a PhD in electrical engineering from Stanford University. Contact him at vwani@ee.ucla.edu.*