Benefits of Targeting in Trusted Gossiping for Peer-to-Peer Information Sharing

Arindam Mitra[1] and Muthucumaru Maheswaran[2]

[1]University of Manitoba
Dept. of Computer Science
Winnipeg, MB R3T 2N2, Canada
arindam@cs.umanitoba.ca

[2]McGill University
School of Computer Science
Montreal, QC H3A 2A7, Canada
maheswar@cs.mcgill.ca

Abstract

In a recent study, we proposed a trusted gossip protocol for rumor resistant information sharing in peer-to-peer networks. While trust aware gossiping significantly reduced the rumor spread on the network, we observed that the random message spraying in trusted gossip creates too many redundant messages increasing the message overhead and error rate. In this paper, we propose a message targeting scheme that can significantly improve the performance of the trusted gossip. Our targeting scheme can be easily implemented in a social network setting. We performed large-scale simulations using traces collected from the Flickr social network and other data sets to estimate the performance of targeting in trusted gossip. Our experiments show that significant performance gains can be achieved.

1 Introduction

The phenomenal success of blog and Wiki servers is renewing the interest in *peer-to-peer* (P2P) information sharing that was tried decades earlier in systems such as *rumor monger* developed for internal use at Apple Computer Inc. [26]. The potential of the P2P model to alleviate censorship and bias in information sharing is one of the prime motivator for this renewed interest.

One of the major mechanisms within an P2P information sharing system is message dissemination. An ideal mechanism for message dissemination should provide scalable and efficient implementation, censorship immunity, fault tolerance, and trustworthiness. Previous studies [7, 11] have shown that gossip-based protocols have all the above properties minus the trustworthiness. In gossip protocols, each node selects a random subset of the nodes known to it (referred to as *next nodes*) and forwards them the messages currently held by it. By randomly selecting the next nodes the protocol tolerates unreliable and uncooperative nodes. Further, by limiting (often to a constant) the number of next nodes gossip achieves very high scalabilities.

Most existing gossip based dissemination schemes address the trust concern by requiring off-line trust relations. Obviously, this solution does not support dynamic networks where nodes can join and leave. In a previous work [17], we proposed a *trusted gossip* protocol that supports dynamic networks. The core idea was to *inline* a message filtering process into the dissemination protocol such that the spread of trusted messages is maximized while the spread of untrusted messages is minimized. As messages spread, each node makes an accept, reject, and forward decisions based on a credibility measure which is computed from the estimates of the origin node's trust and the particular message's trust. While trusted gossip significantly improved upon the normal gossip by disseminating messages according to the preferences of the participating nodes, it incurred significant overhead in terms of number of messages and reviews. In this paper, we address the efficiency of trusted gossip by introducing message *targeting* into the gossip process.

In targeted gossip, each origin node maintains a favorable set of target nodes based on acceptance of its messages. A random subset of the favorable set is always included in the next nodes. In this paper, we show that targeted gossip retains *all* the benefits of trusted gossip while significantly reducing various overheads typically incurred by the gossiping process. We performed extensive simulations using actual traces from social networks such as Flickr to evaluate the performance of the targeted gossip protocol. The results indicate significant benefits are achieved by targeted gossip.

In Section 2, we discuss the system model and assumptions. Section 3 presents the targeted gossip algorithm. Section 4 discusses results from trace driven simulations under representative scenarios. Section 5 examines related literature.

1-4244-0910-1/07/$20.00

2 Assumptions and System Model

In this work, we make the following assumptions regarding the information sharing process: (a) each node in the network has some form of *identity* credentials issued by a certification authority, (b) messages are signed using such credentials so that the origin of a message can be established, (c) above message signing process prevents malicious message tampering, (d) information dissemination is only concerned with spreading a digest of the stories and full content is transferred by other means, and (e) nodes can join and leave the network at any time.

The data messages that carrying information (e.g., advertisements, news stories, blog stories) are referred to as *stories*. The stories that are trusted and accepted by a node are *facts* and others that are untrusted and rejected are *rumors*. A story considered as rumor by one node can be considered as fact by another node (i.e., there is no global criteria for evaluating stories). A node attaches higher level of trust with servers that consistently provide facts.

The trusted gossip algorithm is completely decentralized with each node running a *complete* set of functions that can be divided as: filtering and disseminating. A message that comes into the filter will be classified as one of the three types: new story, review request, or review reply. Each message type will then be handled accordingly. A new story can come from the *originating* node or a *relaying* node. The relaying node is a node that has accepted the story itself and wishes to spread the information further. The incoming story is passed through a node-level filter that makes an accept/reject decision based on the reputation of the story originator. The stories that cannot be decided by the node-level filter pass through a message-level filter. Each story passing through this filter is sent away for review by like minded reviewers. A review request that comes into a node is directed to a user willing to review it unless the story has been already reviewed by the node for a prior request. In that case, the old review is sent to the requesting node. When a node receives all replies for reviews requested, it aggregates the ratings and makes the final decision.

When the system bootstraps, all the nodes have the same default node-level trust. This invokes message-level filtering for each message. The node-level trust can evolve in three different ways: (i) post reviews of the stories accepted by a node, (ii) pre reviews of stories yet to be accepted, and (iii) recommendations on nodes. In this paper, we did not use node-level recommendations.

3 Targeted Gossip Algorithm

The targeted gossip algorithm shares the same basic structure as the trusted gossip algorithm we introduced in [17]. It runs trusted dissemination and trust evaluation routines. The trusted dissemination has components running at the sending and receiving nodes. The receiving nodes run the trust based filtering routines to block messages that they deem to be rumors. Sending nodes runs targeting routines to select the most appropriate targets for its messages. The trust evaluation phase is implemented using an augmented Bayesian framework.

3.1 Trusted Dissemination

In trusted dissemination, a receiver's objective is to admit the most *appropriate* stories using a trust aware filtering process. The trusted filtering process will be considered ideal if it can admit all and only those stories that are regarded as trustworthy by all *post* reviews (reviews after acceptance). Similarly, a sender wants to push its stories to receivers that are most likely to accept and further propagate the stories. Transmissions from a sender that are eventually dropped by the receivers are wasted messages. The objective of *targeting* is to avoid this unnecessary overhead while improving the accuracy of the overall algorithm.

The trust filtering module mentioned Section 2 is used for making the accept/reject decision at the receiver and it works by classifying the incoming stories. The classification process is based on the observation that the trust of a origin node is determined by the perceived quality of the stories it produces. Nodes that have created stories that are consistently highly rated by receiving nodes will have high trust scores. That is, we can expect a story originating from a highly trusted node to be of high quality with high probability. Suppose the receiver's trust is T_{rcvr} and story originator's trust is T_{orig}. Let $0 \leq \varphi \leq 1$ be a trust threshold value set by the receiver. A story is classified as a FACT if $T_{orig} \geq (1 + \varphi)T_{rcvr}$ and is accepted by the receiver node into the local aggregate. If $T_{orig} \leq (1 - \varphi)T_{rcvr}$ the story is tagged as RUMOR and dropped. Other stories are classified by a receiver as QUESTIONABLE and additional information is solicited by sending *review requests* to a random set of nodes. A review request message expects the reviewing node to rate the story according its knowledge. Setting φ high means the receiver will increase its reliance on solicited recommendations than the trust of the originator.

If a user at the reviewing node has the time and knowledge to process the review request, a *review reply* is sent to the requesting node. At least ϕ_1 positive review replies are required for a story to be accepted. Moreover, a receiving node keeps track of the previous review requests posted for a story. If this count exceeds the threshold value of ϕ_2 then the story is discarded. Because of the decentralized architecture, we need to use P2P aggregate estimation process [11] to determine the outstanding review requests. However, in this work, we use a simple approach where a node

remembers review requests that passes through it.

Maintaining Target Lists: We assume that each origin has certain number of *friend* nodes with whom it has solid trust that is supported by offline relations. Although these relations are immune to cheating and other malicious activities, the dissemination process cannot exclusively use these relations. Suppose we restrict the story transmissions to friend nodes, a busy origin's friends could be inundated with messages. In addition, an origin node will only be able to disseminate messages acceptable for its friend nodes.

In our scheme, when the origin disseminates a story, it includes its identity and a sequence number in the story header. When a friend node receives a gossiped story and if the relaying node is not the origin, the friend sends a copy of the story to the origin. The origin can use the sequence number in the header to determine the next node it selected for this particular transmission. The forwarded message from the friend node serves as a confirmation for the origin that the next node has accepted and propagated the story.

We assume that any node within two hops away from a given origin node on the social network can qualify as its friend. However, an origin node does not select all of its social neighbors as friends. It randomly changes its friends within its social neighborhood to reduce the load on them and to prevent malicious nodes from initiating false confirmations. The set of friends who are actively forwarding confirmations is not publicly known.

In addition to the friend nodes, each origin maintains a fixed-size list, called *target list* (TL), of favorable next nodes that are known to have accepted stories from the origin. Besides TL, the origin maintains a *test-TL* that does not have any overlap with TL. When the origin learns about a favorable next node, it adds the node to the test-TL. Each node in the TL and test-TL is assigned a counter c, initialized to 1. Every time the origin receives a confirmation about a node in TL or test-TL, the corresponding counter value is incremented by 1. Based on the counter value, a weight ω is computed for each member in both lists. Only the nodes with the k highest ω values will be in TL and the remaining nodes in test-TL. When a test-TL member's ω grows more than a TL member's weight it replaces the TL member. Weights for TL and test-TL members is computed as: $omega_i = \lambda^{\Delta t_i} \frac{c_i}{\sum_{j=1}^{k} c_j}$ where, i and j are members in the origin's TL, k is number of members in TL, $0 < \lambda < 1$, and Δt_i is the time elapsed since last confirmation for node i.

The weight ω is used for selecting next nodes from target lists. To avoid selecting the same set of next nodes, we randomly select κ nodes from TL (i.e., TRGT) and κ' from test-TL (i.e., testTRGT), and κ'' other nodes (i.e., non-TRGT) from the network. These selections are made subject to $\kappa \geq \kappa' \geq \kappa''$.

3.2 Trust Evaluation: Augmented Bayesian Approach

Our trust evaluation framework is an extension of the Bayesian trust framework in [22]. We briefly describe the Bayesian trust framework before presenting the extensions. Please see [16] for details.

Although the Bayesian framework involves complex computations, it provides a theoretically sound basis for computing reputation scores for the nodes. In this framework, *a posteriori* reputation score is computed by combining *a priori* reputation scores with new rating values [19, 22, 30]. The reputation score is then presented in the form of probability expectation value.

Unlike earlier works, the Bayesian framework in [22] provides rating values that can range over 5 discrete levels. In this framework, each node P_x maintains a *Direct Trust* (DT) table for every origin node P_y from which it receives a story; DT serves the *a priori* value for the Bayesian framework. The DT is a *5-tuple*, $(d_1, \ldots, d_5)$ where d_j is the probability that P_x has a direct trust at level l_j for peer P_y and $1 \leq l_j \leq 5$. The direct trust at level l_j is represented as $p(DT_{x,y} = l_j)$. Node P_x uses the DT to compute P_y's trust as $T_{x,y} = \sum_{j \in [1,5]} p(DT_{x,y} = l_j) \cdot \frac{j}{5}$.

Besides DT, node P_x also maintains an experience table ET to store its experience with P_y; $|ET| = 5 \times 5$ and the cell $e_{\alpha\beta}$ in ET represents the probability that even though P_x has a direct trust in P_y at level l_α (or event 1) it has evaluated (i.e., post reviewed) the new story from P_y at level l_β (or event 2). The values in ET are used to compute the *likelihood*, or condition probabilities used for evaluating the posteriori every time P_x receives a new story, using Bayes' Theorem [9] for *2 events*.

An originating node can create different stories with different expected trust levels. The Bayesian framework in [22] does not support this situation. We extend the framework by incorporating a *pre-rating* from the story origin. We implement the extension by modifying the experience table ET to include the origin's pre-ratings.

Therefore, in our system, ET is an $5 \times 5 \times 5$ table, where the cell $e_{\alpha\beta\gamma}$ represents the probability that P_x having a direct trust at level l_α (or event 1) post-reviews a new story from P_y at level l_β (or event 2) when the pre-rating from P_y for the story was at level l_γ (or event 3). The posteriori is then evaluated using Bayes' Theorem for *3 events* [9]; the likelihood is computed using our augmented ET. The process to evaluate the posteriori at P_x for P_y is as follows.

Before receiving: Before accepting a new story, the receiver node P_x has the following prior belief probabilities about P_y which is expressed by the 5-tuple $(d_1, \ldots d_5)$. The weighted sum of the tuple is used to compute its existing trust level l_α for P_y given as: $l_\alpha = \sum_{j=1}^{5} \frac{j}{5} \cdot p(DT_{x,y} = l_j)$

After receiving: After receiving a new story with pre-

rating at level l_γ, P_x will evaluate the new story and assign it a post-rating of level l_β. Then the likelihood is computed using the ET values as:

$$\Theta = p(ET_{x,y} = l_\beta | DT_{x,y} = l_\alpha) = \frac{1}{5} \sum_{j=1}^{5} \frac{e_{\alpha\beta j}}{\sum_{i=1}^{5} e_{\alpha i j}}$$

$$\Lambda = p(rec = l_\gamma | ET_{x,y} = l_\beta, DT_{x,y} = l_\alpha) = \frac{e_{\alpha\beta\gamma}}{\sum_{j=1}^{n} e_{\alpha\beta j}}$$

where, Θ represents the conditional probability of evaluating a story at level l_β when P_x's direct trust for P_y was at l_β and Λ is the conditional probability for P_y providing a pre-rating at l_γ but P_x has evaluated the story at l_β when its direct trust for P_y was at l_α.

Based on this likelihood P_x's direct trust with P_y is computed using Bayes' theorem as follows: $d_\alpha^t = \frac{d_\alpha^{t-1} \cdot \Theta \cdot \Lambda}{\sum_i d_i^{t-1} \cdot \overline{\Theta}_i \cdot \overline{\Lambda}_i}$ where, $\overline{\Theta}_i = p(ET_{x,y} = l_\beta | DT_{x,y} = l_i)$ and $\overline{\Lambda}_i = p(rec = l_\gamma | ET_{x,y} = l_\beta, DT_{x,y} = l_i)$

After performing post-reviewing of the new story, the node P_x will updates it experience with P_y, i.e., the experience table $ET_{x,y}$ is updated accordingly to reflect P_x's change in trust level for P_y. This is given as: $\forall \alpha \in [1,5]$, $e_{\alpha\beta\gamma} = e_{\alpha\beta\gamma} + d_\alpha^t$ and $\forall \gamma \in [1,5]$, $e_{\alpha\beta\gamma} = d_\alpha^t$.

Currently, our trust evaluation framework does not seek recommendations on node reputations but can be extended to accommodate recommendations in a straightforward manner and doing so will significantly reduce the time required for estimating a node's trust value.

3.3 Analysis of Targeted Gossip

We provide a simple analysis to show scalability and censorship resistance properties of targeted gossip. First, we compute the per node and per round overhead created by targeted gossip and establish it is independent of the network size. This indicates targeted gossip is scalable w.r.t. network size. Second, we sketch a proof that reveals targeted gossip is censorship immune under certain conditions.

The messages created by targeted gossip can be grouped into: (a) gossip messages, (b) review and reply messages, and (c) confirmation messages. Consider an origin node P_o that is gossiping with degree d. Each round it sends out $d = \kappa + \kappa' + \kappa''$ messages. Suppose a fraction $0 < k \leq 1$ of the d receivers consider the story QUESTIONABLE and each such node solicits m reviewers for endorsements and p is the probability that a reviewer provides a review. Then the total overhead from reviewing is $kd(m + mp)$. Also, let P_o have f friend nodes and in a given round let p' be the probability that a friend sends a confirmation to P_o. The message overhead due to confirmation messages is given by $f \times p'$. Therefore, the total overhead is $M = d + kd(m + mp) + fp'$, which is independent of the network size.

For the purposes of this analysis, we define *censorship* as the act where a given set of nodes systematically filter all or portions of messages disseminated by an origin node. The trust based filtering proposed in our study is different from this scenario because trust based filtering is a collaborative process where filtering nodes are implementing the requirements of the community. Conversely, in censorship, censoring nodes try to prevent the community from receiving certain messages that the community would like to receive.

Consider the situation where a single origin P_o is disseminating its stories into the network. The dissemination network can be modeled as a random graph where the vertices are nodes including P_o and an edge denotes that the two corresponding nodes are communicating P_o's messages. We consider two nodes to be communicating P_o's messages if one of them is not filtering the messages for malicious reasons. This does not mean all of P_o's messages will be communicated over the edge – only the ones deemed trustworthy by the nodes will be communicated.

The origin P_o can push its messages to all or most of the nodes in the network without being censored if there is giant connected component in the random graph. If a giant connected component exists, P_o is part of it because the edges denote the transmission of P_o's messages. From random graph theory [20], a giant connected component exists if the number of second order neighbors are more than the number of first order neighbors. Suppose the origin P_o disseminates to d nodes and the targeting mechanism to position has been exploited to place k malicious nodes in the d node gossip set. Then, the dissemination reaches $d(d-k)$ second order neighbors. For a giant component, we should have $d^2 > d + dk$. This means the condition $d > (1 + k)$ should be true for censorship resistant communication from P_o when up to k malicious nodes are in the gossip set.

4 Performance Evaluation

In this section, we discuss the results from simulation studies performed to evaluate performance of targeted gossip. The simulators were written in Python and PARSEC [2]. All experiments discussed below used a network setup with $N = 10K$ ($K = 10^3$) nodes, unless stated otherwise.

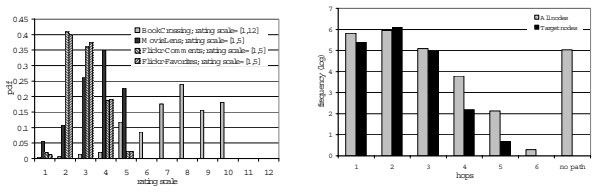
4.1 Trusted Dissemination

Experiment Setup - In our experiments, reputations of nodes are computed as the average of all the ratings for all stories they originate. Story ratings from three different datasets were used to examine the performance of our targeted gossip algorithm. The datasets used were Flickr [6], Bookcrossing [31], and Movielens [12]. Flickr is a popular online photo-sharing service, Bookcrossing is an

online book rating service, and Movielens is an online recommender system for movies. Object rating values from these datasets are used to rate the stories disseminated in our experiments. Figure 1(a) shows the rating distribution for the three datasets used in our experiments, where the x-axis is the rating scale used by the datasets. The rating distribution determines the node reputations used in our experiments. In Flickr, most nodes have low reputation, whereas in Bookcrossing and Movielens most nodes have medium to high reputation. The story ratings from the Flickr dataset were computed in two ways: (i) *Flickr-Comments* - number of comments left by Flickr users for Flickr photos and (ii) *Flickr-Favorites* - number Flickr users declaring a Flickr photo as their favorite. For our experiments, we used three traces with number of stories $M = 51838$ for Flickr, $M = 5647$ for Bookcrossing, and $M = 2346$ for Movielens. Figure 1(b) shows a statistic of the connectivity among users in the Flickr social network which serves as the base network for our experiments. The gray columns show the distances between Flickr users in general while the black columns depicts the distance between story originators and their target-nodes. Our analysis showed the neighborhood graphs formulated by Flickr ratings have power-law distributions.

In our experiments, each node selects the number of targets given by the *gossip degree* (d) such that it is uniformly distributed in [2,5]. The d targets are chosen from TL, test-TL, and the rest of the network as explained previously. The size of TL, test-TL are set at $k \times d$, where k is a parameter selected uniformly from [2,3]. We assume that a node has f *friends* in the network. For certain experiments, the friends were placed randomly in the network and for others the friends were selected from a social neighborhood. We ran the experiments until the gossip process converges, i.e., nodes have no new stories to gossip.

We compare the *targeted* gossip algorithm (TA) with *untargeted* trusted dissemination schemes proposed in [17]. In this paper, targeted and untargeted gossiping are sometimes collectively referred to as trusted gossiping. In the untargeted algorithms, trusted dissemination uses normal gossip with three types of trust-based filtering, described as following. (a) **Receiver-Initiated** (RI) - In this approach, both node and message level filtering mechanisms are implemented at the receiver side. Upon receiving new stories, a node makes a decision based on its knowledge of the story originator's reputation (in our case, reputation is measured by the trust the local node has with the story originator) and/or with the help of the community's review about the story. (b) **Sender-Initiated** (SI) - In this approach, receiving nodes perform the node-level filtering and the sending nodes (origins) perform message-level filtering. technique is The idea is to *prescreen socially unacceptable stories* at the source so as to not spam the whole network. This also



(a) Rating distribution. (b) Distances among nodes.

Figure 1. Properties of input datasets.

helps the source nodes maintain high reputations. (c) **Hybrid** (HY) - This approach combines RI and SI. Receiving nodes perform the node-level filtering while message-level filtering is done probabilistically at both sender or receiver nodes. One of the drawbacks hybrid shares with SI is the limitation in estimating story acceptance. That is, these methods can only target stories for the whole population. They are not suited for targeted dissemination.

Performance Metrics - Following metrics are used to evaluate the performance of the targeted gossip algorithm.

Dissemination Error: measures the failure (error probabilities) of the algorithms; (a) *false negatives* (FN): accepting stories which should not be accepted and (b) *false negatives* (FP): not accepting stories which should be accepted.

Find Capability: measures how many of the target-nodes a node can find during its lifetime. This estimates the discovery capability of the targeted gossip algorithm as the network configuration evolves.

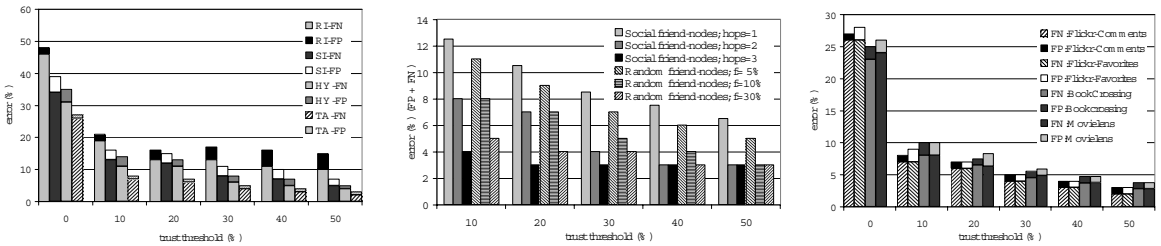
Message Overhead: measures the message count in three parts: (i) *gossip overhead*: the total message overhead for gossiping stories, (b) *review overhead*: the overhead due to review requests and replies, and (c) *TL update overhead*: the overhead due to target-node list updates.

Speedup: is given by $S_{TargetAlgo}/S_{BaseAlgo}$, where $S_{TargetAlgo}$ and $S_{BaseAlgo}$ are the time taken for the facts to reach all nodes using the targeted (i.e., *TargetAlgo*) and normal (i.e., *BaseAlgo*) gossiping algorithms, respectively.

Benefit: measures the improvement in message overhead the targeted algorithm can yield compared to the base algorithm. $Benefit$ is given by $\frac{Overhead_{BaseAlgo} - Overhead_{TargetAlgo}}{Overhead_{BaseAlgo}}$.

Gain: gives the reduction of in user decisions necessary to accept or reject new stories. This metric indicates the savings in users time due to overall collaborations implemented by trusted gossip. Gain is defined as $Gain = \frac{untrusted\ \#decisions - trusted\ \#decisions}{untrusted\ \#decisions}$. In trusted gossip, a user's decisions are leveraged by the community, whereas, in normal gossip a user's decisions are only locally used.

Results and Discussion - Figure 2 shows the variation of dissemination error with trust threshold for all trusted gossip algorithms, TA, RI, SI, and HY. Total error is the sum of false negatives and false positives, shown by the FN and FP components in the graphs. Figure 2 shows the results when



(a) (b) (c)

Figure 2. Dissemination errors: (a) using likeminded reviews , (b) for TA using different friend sets, and (c) for TA using different datasets.

story review requests are sent to only likeminded nodes. We define likeminded nodes as nodes that are within three hops from each other on a social graph (e.g., Flickr social graph). This is in agreement with the trusted friend circles defined in other existing social networks. Moreover, Figure 1(b) shows that majority of the socially connected nodes in the Flickr network are within three hops of each other.

Figure 2(a) compares the trusted gossiping algorithms using the Flickr dataset. When trust threshold $\varphi = 0$, receivers make all accept-reject decisions based on the origin nodes' reputation values. From Figure 1(a) we know that only small number of nodes have high reputations, a large proportions of stories originating at nodes with medium to low reputation will be rejected. Results in Figure 2(a) show that such decisions are erroneous approximately $25-50\%$ of the time. When $\varphi > 0$, node-level reputations are combined with message-level recommender based decisions. As a result, decisions regarding stories considered reputed by the system were made with higher accuracy. Increasing φ meant such combined decisions are made for larger fraction of stories, which improved the accuracy level. In SI, the message-level filtering is done at the origin nodes and the chances of finding good quality reviewers in the simulated Flickr network was high closer to the origin, consequently SI made more accurate message-level decisions, and thus performed better than RI. However, SI suffered from *under-exposure* because stories not sanctioned by the reviewers and dropped at the origin could have been accepted by some nodes. Conversely, the RI algorithm suffered from *over-exposure* because nodes were exposed to stories with low trust that have somehow gathered sufficiently high ratings from at least ζ members in the community, where ζ was more than $\phi_1 = \phi_2 = 5$ stipulated by the algorithm. This is the reason for the dominant FP component in the experimental results. Such errors are suppressed to a large extent in HY by probabilistically combining node and message level filtering. The TA algorithm provided the best performance in terms of dissemination error. The primary reason for this reduction is that targeting yields optimal exposure of the nodes to the stories. This is also the reason that FP component of the errors are low across different trust threshold values. Further, results indicate that HY algorithm is capable of achieving errors closer to TA at high trust thresholds.

Figure 2(b) shows the dissemination error variation using different friend sets. These experiments used two types of friend-nodes: (a) *social friend-nodes* - nodes are located within three hops and (b) *random friend-nodes* - nodes randomly placed over the whole network. Unsurprisingly, results show that larger friend-node sets provide lower dissemination errors. Further, social friend-nodes are more effective in reducing dissemination error than random friend nodes. For instance, for $f \geq 5\%$ (at least 5% of the total nodes are friends) the number of random friends are much higher than the number of social friends within three hops.

Figures 2 (a) and (b) show that combining reputation and recommendation based decisions in targeted gossip results in very low dissemination errors. This is supported by Figure 2(c) which shows the error values for different datasets. As the trust threshold is increased, the error values for the algorithms converge. Figure 2(c) shows targeted gossip being less sensitive to reputation distribution among nodes compared to a simple reputation based system (i.e., when $\varphi = 0$). Error differences among the datasets consistently decrease even though the datasets had varied populations of trusted and untrusted nodes as shown in Figure 1(a).

Figures 3(a), 3(b) and 3(c) show the benefit estimates using Flickr datasets. Figure 3(a) shows benefit variation against φ when message-level filtering is done using likeminded reviewers. The results show that the trusted algorithms create 49%-83% less messages than normal gossiping. Primary reason for this being that not all stories are gossiped to all nodes by the trusted algorithms. Among the tested algorithms, TA provides the maximum benefit. In TA, due to targeting, story dissemination happen mostly among nodes that consider the stories valuable. With targeting schemes that are capable of discovering as much of the favorable nodes, we can expect optimal story dissemination patterns. As a result, TA eliminates lot of redun-

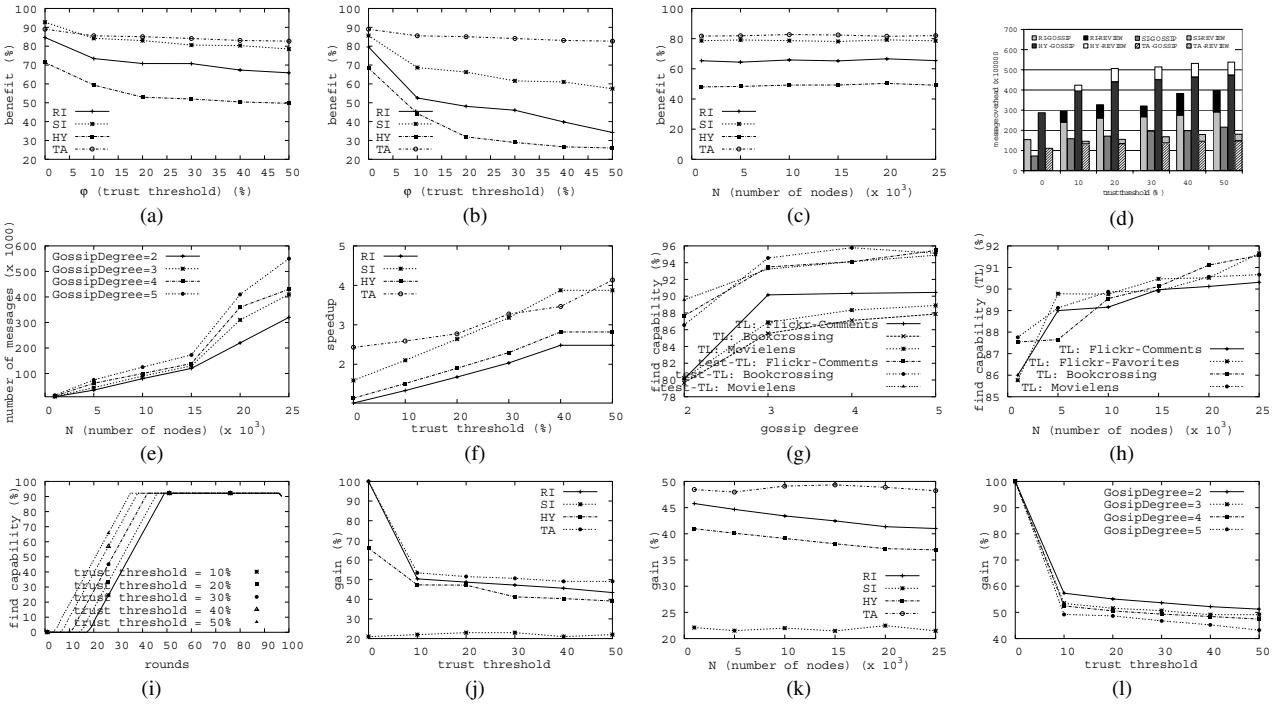


Figure 3. Benefit achieved using likeminded and random reviews are shown in (a) and (b) while (c) illustrates the trusted algorithms as scalable. Gossip message overhead incurred by all algorithms are in (d) while (e) shows the TL update overhead for TA. Speedup estimates are shown in (f). Find capability for TA is shown in (g), (h) and (i) while gain for all algorithms are shown by (j), (k) and (l).

dant messages found in other trusted gossips such as RI, SI, and HY. Among these algorithms, SI causes the least message overhead mainly because this algorithm suffered from under-exposure of the nodes to stories. The over-exposure of nodes in RI leads to much higher message overhead while HY which inherited properties of both SI and RI incurs the maximum overhead. Figure 3(b) shows the results when message level filtering is done using message reviewers from the entire node set, i.e., not limiting to like-minded reviewers within three hops. Comparing Figures 3(a) and 3(b) we see that the benefit for TA does not change much while RI, SI and HY algorithms show reduced benefit. Experimental logs showed that for RI, SI and HY the dissemination error, mainly the FN component, increased by 5%-8% which resulted in higher gossip overhead. Figure 3(c) shows the variation of benefit when the network is scaled up. Scaling the network also increases the number of stories to be gossiped; logs showed an increase by a factor $k = [4, 6]$, i.e., $M = k \times N$. The results show that the message benefit remains approximately constant as network size was scaled. Experiments with other datasets also showed similar scalability trends.

Figure 3(d) shows the gossip and review message overheads for the algorithms. The reviewing overhead is a small fraction of the gossip overhead. Although the simulations gossiped each story separately, an implementation will gather all stories for a given period of time and then gossip the collection as a single large message. Figure 3(e) shows the overhead incurred by TA for keeping the TL set up-to-date. Although this overhead rapidly increases with the network size, it still forms only 0.01% of the overall overhead shown in Figure 3(d).

Figure 3(f) shows the variation of speedup with trust threshold for the trusted gossips. Speedup is basically the ratio of the message spreading times of the trusted and normal gossip. Because story dissemination is lesser in trusted gossip it obviously takes less time. Among the trusted gossip algorithms, SI and TA provide maximum speedup. SI achieves higher speedup at the expense of under-exposing the nodes to the stories. RI has the least speedup due to the over-exposure problem. TA performs best compared to other algorithm due to targeting.

Figures 3(g), 3(h) and 3(i) show the variation of the find capability for TA algorithm. Find capability quantifies the ability to discover the best nodes for the target lists. We compute the find capability as follows. First, using the traces we determine the *ideal* target nodes for each origin. Next, we compute find capability as the percentage of time

one of the ideal target nodes was placed on the TL or test-TL by the targeting algorithm. Figure 3(g) shows that TA is successful in locating more than 80% of the target-nodes.

As mentioned earlier, we used $\kappa > \kappa' > \kappa''$ to decide how many nodes should be chosen from the TL, test-TL, and rest of the network, respectively. For all experiments, we set $\kappa = d/2, \kappa' = d/3, \kappa'' = d - (\kappa + \kappa')$. For experiments with $d = 2$, $\kappa = 1$ while κ' and κ'' was randomly set to 1 or 0 with equal probability. Figure 3(g) shows that the find capability increases sharply as d increases from 2 to 3. This indicates that for targeting it is necessary to maintain some level of random selection in the dissemination process. The random selections allow the dissemination process to spread the stories to a wider set of nodes. With large gossip degree values, certain number of targets are selected from the TL and test-TL sets and others are randomly selected. Figure 3(h) shows the variation of find capability with network size. The results indicate that TA is highly scalable and adaptable to network expansions. Figure 3(i) shows evolution of find capability with time. The results demonstrate the targeting process undergoes a learning phase and the length of the learning phase depends on the trust threshold.

Figures 3(j), 3(k) and 3(l) show the variation of gain using the Flickr dataset. Using normal gossip, each node makes independent decisions for each story and these decisions are never shared with the community. Therefore, with M stories and N nodes, total number of user decisions amount to $N \times M$. In trusted gossip, we want to share the decisions among likeminded users along with the node reputations to reduce the number of decisions made by the users. Further, each story is reviewed only once by a user but a story can receive multiple reviews from different users. Figures 3(j) and 3(k) show that SI achieves a constant 20% gain independent of the trust threshold and the number of nodes, respectively. In our experiments, each origin uses a fixed number of reviewers to pre-review a story before it releases the story into the network. Therefore, SI's gain is indicative of the performance of a recommender based system. In RI, at threshold $\varphi = 0$, nodes make decisions based on reputations without requesting any reviews for the stories. At $\varphi > 0$, reviews are requested by the receiver nodes in RI. Therefore, increasing φ reduces the gain because review requests will be sent out for a larger fraction of the stories. As can be noted from the results, the decrease in gain for increases in threshold in small. We attribute this to the fixed size of the reviewer set. The HY algorithm initiates the review process at the sender and receiver sides probabilistically and it performs better than SI. Further, the results indicate that RI and HY which combines reputation and recommender based approaches can almost double the gain of a pure recommender based approach.

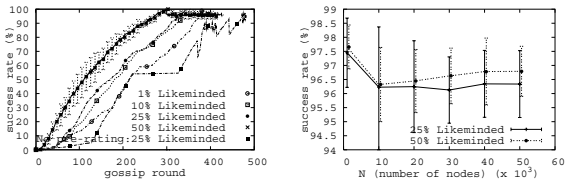
Figure 3(k) shows that the TA algorithm achieves 2% to 8% improvement over RI in terms of gain. The RI algorithm has the best gain value among all the untargeted trusted gossips. We attribute this improvement in gain to the best node-to-story exposure provided by TA. Specifically, because TA is able to cut down the over-exposures that can lead to redundant review requests that can create more overheads. Figure 3(k) shows the the gain values for TA and SI remain almost constant as the network size is scaled up while the gain for RI and HY show small decreases. Because the decreases on the gain of RI and HY are small compared to the network increases, we can claim that all algorithms are scalable. Figure 3(l) shows the variation of gain with the gossip degree d for TA. As we increase d, stories gain more exposure that creates more review requests, which leads to lower gain.

4.2 Trust Evaluation

Experimental Setup - Receiver nodes compute trust values for originating nodes. In the experiments below, we evaluate the trust values every gossip round. We associate an error probability with each node which when combined with a story's ratings is used to denote the story's quality as evaluated by that node (i.e., post review). Likeminded nodes will have similar error probabilities.

Results and Discussion - Figure 4(a) shows the success rates for computing trust values by our augmented Bayesian framework. *Success rate* is the percentage of the nodes whose behavior the framework is able to predict correctly in a round. The figure shows the success rates of our framework and compares it with the existing (or base) framework [22] where no pre-ratings are provided by the story origin. Both frameworks were able to estimate trust values with equal accuracy, but clearly, the time taken to compute the trust correctly is much less in the augmented framework. Primary reason for this being that more information is used to compute a node's trust. The result shows that even with a low number of likeminded nodes, the augmented framework is able to converge faster than the base framework.

Figure 4(a) also shows that within the augmented framework, as the number likeminded nodes increase the time taken to estimate the trust values is reduced. This was expected since likeminded nodes will be evaluating each others trust more accurately with each story exchange. The error-bars in the figure shows that the error estimation process improves with every subsequent round. It should be noted, that we did not include node recommendations in our experiments. We conjecture that using node-level recommendations to compute trust will improve the time taken for trust estimation. Without node-level recommendations, the time estimates from our trust evaluation framework cannot be compared to existing systems that use active polling to collect community information to compute trust.



(a) Augmented Vs Base model. (b) Scalability of TA.

Figure 4. Trust computation using 10K nodes.

Figure 4(b) illustrates the scalability of the framework when the number of nodes and stories were scaled up. Here nodes do not over-rate their story's' qualities. The results show that with more likeminded modes the success rate increases by a small amount. The average success rate falls slightly when network size increases from $1K$ to $10K$. But for larger networks the success rate remains almost the same. The error-bars show that with larger network or with more likeminded nodes the accuracy improves by a very small amount. Thus, the augmented approach exhibits scalability to a reasonable extent.

5 Related Work

Information Dissemination Systems - *P2P file sharing networks* (PPFNs) [25] and *content delivery networks* (CDNs) [29] are popular dissemination systems used for pushing content across geographically distributed networks. PPFNs and CDNs implement integrity checks by merely verifying the content as authentic, whereas, targeted gossip deals with a broader notion of *appropriateness* of the stories. In systems using targeted gossip, users seek information that is previously unknown to them. By using a combination of server and message level trusts evaluations, our proposed system can present an unbiased collection of stories to the users while preventing rumors from spamming the network. *Usenet* [27] another popular system, employs techniques like collaborative filtering [12] to reduce users' immediate visibility to small chosen set of articles but does not prevent the spreading of unwanted articles in the network. Publication systems like blogs and Wikis disseminate via *hot linking*, external means (e.g., emails) or by *syndication* [10, 21] where users selectively pull news item(s) of interest. In contrast, our dissemination model is a completely distributed push-based system using a inlined filter to prevent spamming with less trustworthy stories.

Rumor spreading in large networks is similar to computer virus spreading. Current studies [4, 8, 18] modeling viral propagation using epidemiological models consider a virus as harmful at all nodes in the system; a virus will be treated same way by all nodes. This approach is in absolute contrast with ours where a story considered as rumor or spam by one node can be considered as fact by another node. Meaning, stories are treated differently at different points of the network. Further, targeted gossip can be considered as a form of regional gossip where the regions are defined based on message interest. This is in contrast to gossiping schemes [5, 7, 14] based on node densities or lacking message trust information.

Reputation/Recommender Based Systems - Reputation systems [13, 24] compute global estimate of a node's trust solely based on a node's behavior without involving the quality of its messages/transactions. This effects overall accuracy since it becomes difficult to distinguish trustworthy transactions from less reputed nodes and vice versa. In contrast, our proposed system defines a trust spectrum to classify nodes, where a node's reputation is a definitive indicator if it is placed very high or very low in the spectrum; for all other cases, we combine the message level recommendations to make the decisions. Recommender systems [1, 23] also provide global context of users' behavior. Models [12, 28] proposed for this purpose use user ratings for news/products to make predictions for likeminded users or employ social data mining techniques [15]. In a distributed environment, such systems result in very high network message overhead by active polling of recommendations. Our simulations show that our model is able to contain the explosion of recommendation messages by combining recommendations with reputation based decisions.

Trust Evaluation Systems - Trust based systems have been widely employed to create cooperative distributive systems. In [19], a Bayesian formalization for a distributed rating process is proposed. However, this work considers only binary ratings and does not decay the rating values over time. In the Bayesian reputation scheme presented in [3], the aging of ratings were considered but still it used binary ratings. The binary ratings issue was tackled in [30] which used a rating scale in range [0, 1] to describe a Bayesian approach to filter unfair ratings. The framework [22] provides further flexibility of using a n-level rating scheme. It provides lightweight n-level trust metric which protects user anonymity. We extend the Bayesian framework in [22] to make the trust metric more expressive; we incorporate more information about a node's behavior. Results show that our model was able to compute a node's trust quicker.

6 Conclusions

In this paper, we proposed a targeted gossip algorithm where an inline filtering process is used while gossiping to retard the spread of unacceptable stories and accelerate the spread of acceptable stories. The targeted gossip includes a targeting phase which is used by each story originator to determine the best audience for its stories.

We performed extensive simulations using traces ob-

tained from the Flickr social network and other datasets to verify the performance of targeted gossip. Our results indicate that targeting significantly increases the accuracy of trusted gossiping while the message complexity and user engagement are simultaneously reduced. The reduction in user engagement is particularly compelling because the total user time spent on reviewing stories is reduced by cutting down the redundant transmissions. In the targeted gossip, each origin uses a set of friends to provide feedback on the dissemination process. The simulation results indicate that selecting the friend nodes from the two-hop social neighborhood is sufficient. This friend node selection strategy is practical in almost all social networks.

In conclusion, we developed a targeted gossip algorithm and established that it provides significant benefits using traces from actual social networks. The targeted gossip can be used as a building block for a new class of P2P information sharing applications such as P2P news networks and social advertisement dissemination networks.

References

[1] A. Abdul-Rahman and S. Hailes. Using Recommendations for Managing Trust in Distributed Systems. In *IEEE Malaysia Intl. Conference on Communication*, Nov. 1997.

[2] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, B. Park, and H. Song. Parsec: A Parallel Simulation Environment for Complex Systems. *IEEE Computer*, 31(10):77–85, Oct. 1998.

[3] S. Buchegger and J. Boudec. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *2nd Wksp. on Economics of Peer-to-Peer Systems*, Jun. 2004.

[4] Z. Chen, L. Gao, and K. Kwiat. Modeling The Spread of Active Worms. In *IEEE Infocom*, Mar. 2003.

[5] P. Eugster, R. Guerraoui, S. Handurukande, A.-M. Kermarrec, and P. Kouznetsov. Lightweight Probabilistic Broadcast. *ACM Trans. on Comp. Sys.*, 21(4):341–374, Nov. 2003.

[6] Flickr. http://www.flickr.com.

[7] A. Ganesh, A. Kermarrec, and L. Massoulie. Peer-to-Peer Membership Management for Gossip-based Protocols. *IEEE Trans. Comp.*, 52(2):139–149, Feb. 2003.

[8] M. Garetto, W. Gong, and D. Towsley. Modeling Malware Spreading Dynamics. In *IEEE Infocom*, Mar. 2003.

[9] P. Gregory. *Bayesian Logical Data Analysis for the Physical Sciences: A Comparative Approach with Mathematica Support*. Cambridge University Press, Cambridge, UK, 2005.

[10] T. Hammond, T. Hannay, and B. Lund. The Role of RSS in Science Publishing. *D-Lib Magazine*, 10(12), Dec. 2004.

[11] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based Computation of Aggregate Information. In *44th Annual IEEE Symposium on Foundations of Computer Science*, Oct. 2003.

[12] J. Konstan, B. Miller, D. Maltz, and J. Herlocker. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, Mar. 1997.

[13] C. Lampe and P. Resnick. Slash(dot) and Burn: Distributed Moderation in a Large Online Conversation Space. In *ACM SIGCHI Conf. on Human Factors in Comp. Sys.*, Apr. 2004.

[14] X. Li, K. Moaveninejad, and O. Frieder. Regional Gossip Routing for Wireless Ad-hoc Networks. In *Mobile Networks and Applications*, Feb. 2005.

[15] H. Liu and P. Maes. InterestMap: Harvesting Social Network Profiles for Recommendations. In *Wksp. in Intl. Conf. on Intelligent User Interfaces*, Jan. 2005.

[16] A. Mitra. *Trust Driven Communication System for Peer-to-Peer Social Networks*. PhD thesis, Dept. of Computer Science, University of Manitoba (Submitted Feb. 2007).

[17] A. Mitra and M. Maheswaran. Trusted Gossip: A Rumor Resistant Dissemination Mechanism for Peer-to-Peer Information Sharing. In *Advanced Information Networking and Applications (Submitted)*, May 2007.

[18] D. Moore, C. Shannon, G. M. Voelker, and S. Savage. Internet Quarantine: Requirements for Containing Self-propagating Code. In *IEEE Infocom*, Mar. 2003.

[19] L. Mui, M. Mohtashemi, C. Ang, P. Szolovits, and A. Halberstadt. Ratings in Distributed Systems: A Bayesian Approach. In *Information Tech. and Systems Wksp.*, Dec. 2001.

[20] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random Graphs with Arbitrary Degree Distributions and their Applications. *Physical Review*, 40, Jul 2001.

[21] M. Nottingham and R. Sayre. The Atom Syndication Format, IETF Internet Draft, Jun. 2005.

[22] D. Quercia, S. Hailes, and L. Capra. B-trust: Bayesian Trust Framework for Pervasive Computing. In *iTrust*, May 2006.

[23] P. Resnick and H. R. Varian. Recommender Systems. *Comm. of ACM*, 40(3):56–58, Mar. 1997.

[24] P. Resnick and R. Zeckhauser. Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System. In *Advances in Applied Microeco.*, Feb. 2002.

[25] M. Ripeanu. Peer-to-Peer Architecture Case Study: Gnutella Network. In *1st Intl. Conf. on Peer-to-Peer Computing*, Aug. 2001.

[26] RumorMonger. http://apple.computerhistory.org/.

[27] R. Salz. InterNetNews: Usenet Transport for Internet Sites. In *USENIX*, Jun. 1992.

[28] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based Collaborative Filtering Recommendation Algorithms. In *10th Intl. World Wide Web Conference*, May 2001.

[29] A. Vakali and G. Pallis. Content Delivery Networks: Status and Trends. *IEEE Internet Computing*, 7(6):68–74, 2003.

[30] A. Whitby, A. Jsang, and J. Indulska. Filtering Out Unfair Ratings in Bayesian Reputation Systems. In *Wksp on Trust in Agent Societies*, Jul. 2004.

[31] C. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving Recommendation Lists Through Topic Diversification. In *14th Intl. World Wide Web Conference*, May 2005.