AN ANYTIME ALGORITHM FOR TRUST PROPAGATION IN SOCIAL NETWORKS

by

Andrew Johnson Hamilton

A professional project submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

August 2011

APPROVAL

of a professional project submitted by

Andrew Johnson Hamilton

This professional project report has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citation, bibliographic style, and consistency and is ready for submission to The Graduate School.

John Sheppard

Approved for the Department of Computer Science

John Paxton

Approved for The Graduate School

Dr. Carl A. Fox

STATEMENT OF PERMISSION TO USE

In presenting this professional project report in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library.

If I have indicated my intention to copyright this project report by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extended quotation from or reproduction of this thesis in whole or in parts may be granted only by the copyright holder.

Andrew Johnson Hamilton

August 2011

ACKNOWLEDGEMENTS

I wish to thank Hasari Tosun for the initial MRFTrust code and the advice. I also wish to thank Dr. John Sheppard for his advice, his insight, and his patience.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

LIST OF FIGURES CONTINUED

Figure Page

ABSTRACT

Trust propagation in social networks is a challenging task. It is difficult to model human trust, and the data is huge and very sparse. Due to these challenges, the algorithms available to propagate trust have complexity issues. We used the MRFTrust algorithm created by Tosun and Sheppard to produce an anytime algorithm for trust propagation. To do this we use sampling techniques and increased horizon size to reduce the complexity and decrease runtimes. We show that we can dramatically reduce the number of nodes considered in the algorithm, yet still achieve a superior result.

INTRODUCTION

In the digital age, we are increasingly an "online" society. Where once we were confined to email, a little web surfing and for the braver among us, the occasional digital purchase, we now live a fair amount of our lives online. While our internet usage has grown exponentially, the technologies we use to navigate the somewhat dangerous and un-trusted jungle remain mostly the same. There have been some incremental improvements in narrowly defined segments of security and access. The level of trust we have is an ever-growing gap, and it is difficult to authenticate where we go and what we trust. Sure we can use SSL to secure our transactions and communications, but that does not help with being able to differentiate between who we are encrypting with. Even businesses have trouble with trust policies and most of these policies can be easily breached [1]. It is quite common to interact with unknown "persons" on the web. For instance when interacting with a stranger via email, when exchanging messages on a web site and reading reviews of products or articles written by someone we neither know nor have methods to determine if they can be trusted. Therefore, finding a method for trusting content on the internet is of paramount importance. Phishing scams can be eliminated by computing the trustworthiness of the parties. Ecommerce sites can verify the trustworthiness of their customers and partners. Political organizations can be extended to strangers without the fear of government repression or persecution [2].

Due to the fact that these networks are sparse and massive, available algorithm performance tends to be slow. It is necessary to introduce techniques that will assist in improving these performance issues. A technique that we explore here is the use of an

anytime algorithm. This will mitigate some of these problems by allowing the output of the algorithm to reach an initial value whose quality may be sufficient for some purposes, and by providing it additional resources, the quality of the output increases.

Trust

Ziegler and Lausen categorized trust metrics on three dimensions [3]: a network perspective, computation locus, and link evaluation. For the network perspective, they categorized trust metrics as global and local. Global trust metrics consider all links and nodes in the networks, where local trust metrics take into account only a partial network. Computational locus refers to the place where trust relationships are computed. In determining distributed or local trust metrics, the computation load is distributed to every node in the network, whereas in computationally centralized systems, all metric computations are performed on a single machine. The tradeoff between the two approaches involves memory, performance, and security. Finally, link evaluation determines whether the trust metrics themselves are scalar or group trust metrics. In group trust metrics, trust scores are computed for a set of individuals whereas for scalar metrics, only trust scores between two individuals are computed. A global trust value is called “reputation” and “reputation systems” [4] are what we call “global trust metrics”. For example, the well known auctions web site eBay.com shows for every user her “reputation” by allowing them to see how many other users left positive, neutral or negative feedback about that user. This “reputation” value is independent from the browsing user and hence we can say that eBay.com uses a global trust metric. To be able

to use existing trust information to infer new trust relationships is a promising approach. Google's page rank algorithm aggregates a "reputation" by recursively computing the authoritative trust–worthiness of each page using the links pointing to that page, and weighting them according to the pages that point to it [5].

These types of global approaches have their shortcomings. Trust is not necessarily a global attribute, which diminishes the value of the global approach. Trust is often subjective and relevant to each individual. But that also does not mean that trust cannot be gleaned totally from personal information. It can be "influenced" by a trusted authority. Yet different entities, henceforth referred to as "users", can be trusted by some and not by others. In this case, it can be difficult to determine if a user can be trusted with any sort of objectivity. Distinguishing between a global value and a local value for a given user can be more accurate in terms of trust. A global value is given to the user regardless of the evaluating user and that can be misleading, yet a local metric can also be a limiting factor.

It is determining how "local" a local network is that we discuss here. Our goal is to determine what the balance point is between a global metric and a local metric to determine if more accuracy can be gleaned from the evidence at hand.

As the Semantic Web gains acceptance, understanding the credibility of metadata about authors is becoming important [6]. While designing recommender systems, one researcher found that there is a strong correlation between trust and user similarity [5]. Thus, trust became the essential variable in computing user similarity [7]. Finally, there is a wealth of information on trust and reputation scoring in social networks [6],[7],[8], [9],

[10]. For example, Mui documented the theories and approaches about trust scores and reputation systems using Bayesian networks for inference on social networks [11]. Barbalet characterized trust and its consequences in detail. He postulates that it is insufficient to define trust in terms of "confident expectation regarding another's behavior" as many researchers defined [4]. Instead, the author characterizes trust in terms of acceptance of dependency (the trust giver grants control or power to trustee; thus, the trust giver accepts dependence on trustee) in the absence of information about the other's reliability in order to create an outcome otherwise unavailable. Golbeck and Hendler adopted a narrower definition of trust for social networks—"trust in a person is a commitment to an action based on belief that the future actions of that person will lead to a good outcome" [10]. Thus, researchers have often proposed simplistic approaches for trust propagation.

Although, researchers generally do not agree on the definition of trust, two properties of trust are used for aggregation: transitivity and asymmetry. Transitivity means if A trusts B who trusts C, then, A trusts C. Asymmetry means if A trusts B, it does not mean that B will also trust A. The majority of trust propagation algorithms utilize the transitivity property [3], [6], [8], [12], [13]. It should be noted this property may not always work with distrust [14]. Moreover, [15] and [16] defined two types of trusts: referreral trust and direct functional trust. If A trusts B who trusts C, then, the trust between A-B and B-C is direct functional trust. However, if B recommends C to A as trustworthy, it is referral trust. Modeling trust networks and propagating trust is a

challenging task: 1) trust networks are huge and sparse, and 2) it is often difficult to model human belief and trust.

An increasing number of articles have been published on modeling trust networks and evaluating trust metrics [3], [4], [10], [12], [14], [15], [17], [18], [19] using different computational methods. For example, Wang and Vassileva designed a Bayesian Network-based trust model for Peer-to-Peer (P2P) communication [19]. The model represents different features of trust as leaf nodes of Naive Bayes networks. On the other hand, [18] developed a model based on Fuzzy logic.

Anytime Algorithms

Anytime algorithms are algorithms whose quality of results increases gradually as computational time increases. The anytime algorithm term was coined by Dean and Boddy in the mid-1980's [20]. According to [21] there are 4 axioms that describe the behavior of anytime algorithms:

1. Initial behavior: The initial period during which the behavior of the method is constant. Many anytime algorithms start producing some output immediately, but other methods need an initial startup period before they start producing intermediate output. We refer to this as "burn-in".
2. Growth direction: This is the direction in which the quality of the intermediate output changes with increasing runtime.
3. Growth rate: The amount of increase in quality at each step during the computation. This increase in quality can be constant at each step, but may also vary during the computation.

4. End condition: The amount of runtime needed for the method to achieve its full (i.e. traditional) functionality. After this point, the quality of the output no longer increases, since the maximum desired quality has been achieved.

In [21], Zilberstein describes several desired properties of anytime algorithms. The first is measurable quality. The quality of any approximation can be measured precisely. For example, the quality reflects the distance between the approximate result and the correct result; it is measurable as long as the correct result can be determined. Next is recognizable quality. This is the ability to determine the quality of the approximate result at runtime. Monotonicity of the result is a nondecreasing function of time and input quality. This is followed by consistency which is a correlation between computation time and input quality. In general, algorithms do not guarantee a deterministic output quality for a given amount of time, but should in general have a narrow variance so that quality prediction can be performed. Another property he describes is diminishing returns, the improvement of the solution quality is larger at the early stages and it diminishes over time. An important property and one that gives a great deal of value to anytime algorithms is interruptability. The algorithm can be stopped at anytime and provide some answer. Preemptability is a property that is desirable since it allows the stopping and restarting of the algorithm with minimal overhead.

Zilberstein goes on to define what he refers to as performance profiles (PP) and conditional performance profiles (CPP). A PP of an anytime algorithm, $Q(t)$, denotes the expected quality with execution time t, while a CPP, $Pr(q_{out}|q_{in},t)$, denotes the probability

of getting a solution of quality, q_{out}, when the algorithm is activated with input of quality q_{in} and execution time t.

Computation in a world of bounded resources is often associated with cost/benefit tradeoffs. With a computational tradeoff, the benefit associated with an increase in the quantity of one or more desired attributes of computational value is intrinsically linked to costs incurred through changes imposed on other attributes. More specifically, we define a tradeoff as a relationship among two attributes, such as the immediacy and accuracy of a computational result, each having a positive influence on the perceived total value of the computer performance, such that they are each constrained to be a monotonically decreasing function of the other over some relevant range. A representative function might look something like:

$$Accuracy = F(Immediacy), t_0 \leq Immediacy \leq t_n$$

where F is some monotonically non-decreasing function over the range of computational time delays t_0 and t_n. The tradeoff between the immediacy and the accuracy or precision of a solution is particularly explicit in methods that incrementally refine a computational result with time [22].

RELATED WORK

Tosun and Sheppard [23] demonstrated that incorporating evidence into a trust model improves the results of trust prediction. Wang and Vassileva [19] show that a Bayesian network based on trust and reputation, sharing trust, can perform better than agent trust alone in P2P networks. The model represents different features of trust as leaf nodes of Naive Bayesian networks. Their proposal of a Bayesian Network works well in a small network where the interactions between the agents are close knit. However, in large sparse networks where the interactions are sparse, their method breaks down. Orman [20] defines trust as a conditional probability and uses that to formulate the problem as a Bayesian Network. In this way he uses well established algorithms and techniques to personalize the trust prediction. Another popular trust model is the Appleseed Trust metric based on a Spreading Activation Model [3] in psychology for evaluating trust in the Semantic Web. The main constituents of their model for Semantic Web trust infrastructure are the agent set $V = a_1 \ldots a_n$ where every agent $a \in V$ is assumed to be uniquely identifiable and a partial trust function set $T = W_{a_1} \ldots W_{a_n}$ which is publicly accessible for any agent in the system. In the latter, every agent *a* is associated with one partial trust function $W_a : V \rightarrow [0,1]^{\perp}$ corresponding to the set of trust association that *a* has stated. Due to the fact that the number of individuals for whom agent *a* has assigned explicit trust values is much smaller than the overall number of agents in the system, these functions will be sparse:

$$W_{a_i}(a_j) = \begin{cases} p, & \text{if } trust(a_i, a_j) = p \\ \perp, & \text{if no rating for } a_j \text{ from } a_i \end{cases}$$

where p is the trust value, and $\perp$ is used if no value exists. This does not indicate untrustworthiness. The higher the values of $W_{ai}(a_j)$ the more trustworthy a_i deems a_j; a value of 0 on the other hand means that a_i consider a_j as not trustworthy at all. The two constituents of the trust model define together a directed trust graph with nodes being represented by agents $a \in V$ and directed $edges(a_i,a_j) \in E \subseteq VxV$ from nodes a_i to nodes a_j being trust statements with weight $W_{ai}(a_j)$. To start a search, source node s is activated through an injection of energy e. To propagate e to other nodes along the edges, e is completely divided among the successor nodes with respect to their normalized local edge weight. To avoid endless, marginal and negligible flow, energy streaming into a node must exceed a threshold T in order not to run dry. Trying to interpret this basic intuition behind the spreading activation models in terms of trust computation give rise to some problems: All energy that has passed through a node x will be accumulated representing its rank but at the same time, all energy contributing to the rank of x is passed without loss to its successor nodes. Massa and Avesani studied challenges of computing trust metrics in a social network where data are sparse [9]. In such networks, neither a global reputation nor a simple trust score is a viable option since a large percentage of the participants are considered to be controversial; they are distrusted by some and trusted by others. Thus, the authors proposed a state-of-the-art framework and algorithm called MoleTrust that uses local trust metrics. However, this approach does not incorporate other sources of evidence. For example, the epinion.com dataset contains articles written by users [17] where these articles are also rated by other users. As far as

we know there is no other anytime algorithm that provides trust propagation in social networks.

METHODS AND PROCEDURES

Trust Metrics

First, we define global and local trust metrics. Formally they are given by:

$$T : U \rightarrow \{0,1\}$$

where U is the trust value of the user, while a local trust metric is given by

$$T : U \times U \rightarrow \{0,1\}$$

Global trust metrics can be computed just once for every user and that value remains the same regardless of the user that is inspecting that value. Those who use theses values assume that this value is objective and the goal is to guess the right number. But different users may have different opinions of the same user; therefore while one person may trust that user, another user might not. This is the heart of the controversiality measure which we define next. Controversiality is the measure of a user's trust and distrust given as:

$$c(x_j) = \frac{|Trust(x_j)| - |Distrust(x_j)|}{|Trust(x_j)| + |Distrust(x_j)|} \qquad (1)$$

where $Trust(x_j)$ is the set of trust statements for user/node x_j, and $Distrust(x_j)$ is the set of distrust statements for x_j. The controversiality level has the range of -1.0 to 1.0. A user with a controversiality of -1.0 is distrusted by all their judges, whereas a user with a controversiality of 1.0 is trusted by all who voted. The users with the most controversiality are the ones at 0.0 as they have an equal number of trust and distrust statements. Next, horizon in our context is the maximum distance from the source user to which trust is propagated along trust chains. This regulates the number of users visited allowing for more control over the computational complexity. A low trust horizon will

reduce coverage, while a high trust horizon will increase the computational time. Coverage is the percentage of users that each of the algorithms can predict in the given controversiality interval. Finally, we refer to "users" as any entity that can be trusted, such as people, web pages, etc.

Markov Random Fields

We start by giving a brief discussion of our algorithm, AT-MRFTrust. The algorithm is based on Markov Random Fields (MRF's). A detailed discussion of MRF's is given by Koller and Friedman [24]. An MRF is an undirected graphical model in which each node corresponds to a random variable or a collection of random variables, and the edges identify conditional dependencies. They have a wide range of application domains. We exploit this model for propagating the trust scores in a local network. The joint probability distribution over X and Y can be represented by an MRF in the following way:

$$P(X,Y) = \frac{1}{Z} \prod_{i,\,j} \psi(x_i, x_j) \prod_{i} \phi(x_i, y_i)$$

where Z is a normalization factor, also known as a partition function, $\psi(x_i,x_j)$ represents pair-wise influence between node x_i and node x_j in the network (often referred to as a compatibility matrix), and $\varphi(x_i,y_i)$ is a local evidence function that forms a distribution over possible states, x_i, given only its observations y_i. This framework allows us to evaluate an active user's trust for an unknown person in the network.

There are two common methods for inference in MRF models [24]: 1) Markov Chain Monte Carlo (MCMC) sampling methods, such as Gibbs sampling, and 2) Belief

Propagation. MRFTrust [23] is based on the second method, Belief Propagation, so we will describe the basic steps in the Belief Propagation algorithm:

1. Select random neighboring nodes x_k, x_j
2. Send message M_j^k from x_k to x_j
3. Update the belief about the marginal distribution at node x_j
4. Go to step 1 until convergence

Message passing in step 2 is carried out as:

$$M_j^k = \sum_{x_k} \psi(x_k, x_j) b(x_k) \qquad (2)$$

where $b(x_k)$ is the current belief value associated with the node. Belief updating in step 3 is then computed thus:

$$b(x_j) = \kappa \phi(x_j, y_j) \prod_{k \in N(j)} M_j^k \qquad (3)$$

where κ is a normalization factor and $N(j)$ is the set of nodes adjacent to node x_j. To infer the trust score of users unknown to the current user in that network, a local network is generated from the global social network. The local trust network has a limited horizon, which is, instead of propagating trust statements using the global network, a local network is created that is specific to the user's neighborhood. As an example, for user *A*, we generate a local network that contains all adjacent users that are only a finite distance, based on the number of links crossed, away from *A*. Thus, the trust score is evaluated relative to the active user *A*. We compared our trust propagation results to the MoleTrust Algorithm presented in [9]. The process we use to generate the local network is similar to MoleTrust in that it is based on the intuition that the average trust path length between two individuals is relatively small [3].

In MRFTrust, the initial belief for the neighbor nodes of the target node in equation 2 is calculated as:

$$b(x_k) = \begin{cases} T(x_k, Target) & Link\ Exists \\ \phi(x_k, y_k) & Otherwise \end{cases}$$

Where $\phi(x_k, y_k)$ is node *k*'s observation about the target node, calculated as:

$$\phi(x_k, y_k) = \begin{bmatrix} \theta \\ 1-\theta \end{bmatrix}$$

and θ is the average rating issued by x_k on articles or documents written by the target user, normalized by the maximum rating for the articles.

MoleTrust

For fairness in our experiments we re-implemented the MoleTrust algorithm. This ensured that our experiments were carefully controlled and our comparisons were valid. To predict how much a user *A* trusts a user *B*, denoted *T*(*A*,*B*), MoleTrust generates a local directed graph from a given global social network whose root is *A*. For each graph depth, it adds links that represent trust statements between users. To avoid cycles, it does not add nodes if they are already in the local network. The depth or distance of the graph is determined by the horizon. If the target user is in the local graph a prediction can be made. Otherwise a prediction cannot be made. This restriction, as we will see, makes a significant impact on the algorithm coverage. Trust propagates from the root node to the leaves with equation 4, where $b(x_j)$ is the trust or belief value predicted at node x_j:

$$b(x_j) = \frac{\sum_{k \in P(j)} b(x_k) T(x_k, x_j)}{\sum_{k \in P(j)} b(x_k)} \qquad (4)$$

Here $T(x_k, x_j)$ is the trust value on the edge between node x_k and x_j, and *P(j)* is the set of nodes with edges terminating at x_j.

The trust values for the nodes are calculated from this equation, where as the trust values on the edges are specified explicitly in the network. Edge trust represents the explicit trust voting of one user about another user. The belief propagation is initialized at 1.0 for the root node.

The MoleTrust algorithm accepts an incoming link to node x_j if the predicted trust value of x_j's corresponding parent node is above a threshold. Otherwise, the link between them is blocked from propagating the trust scores. We claim this approach biases the performance of the MoleTrust algorithm by limiting its predictions to those that are simple. Although this approach may result in accurate trust predictions, it results in low coverage. For a given graph depth or horizon, if the user is not in the local network, MoleTrust cannot make a prediction. Moreover, if trust propagation does not reach the target user due to the link blocking explained above, the prediction cannot be made. For example, if we were to predict Alice's trust in Mark based on the network in Figure 1, without a direct link between them, the trust would have to propagate indirectly through nodes Bob and Dave, however, if neither had a direct link to Mark, a prediction could not be made by MoleTrust.

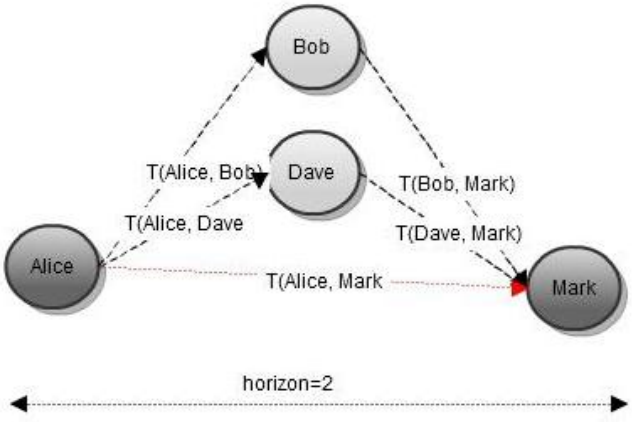

Figure 1 Determining Coverage in Moletrust

The other situation arises when all the parent nodes of the target node Mark, namely Bob and Dave, have the calculated trust score below the given threshold. Since they will be blocked from propagating their trust scores, once again MoleTrust will be prevented from making a prediction. MoleTrust takes no other evidence into consideration. Figure 2 shows the pseudo code for the MoleTrust algorithm.

Step 1:

Input: source_user, trust_net, trust_prop_horizon
dist = 0;
users[dist] = source_user
while (dist ≤ trust_prop_horizon) do
dist + +
users[dist]=users reachable from users[dist ≤ 1] and not yet visited
keep edges from users[dist ≤ 1] to users[dist]

Step 2:

Output: trust scores for users
dist = 0;
trust(source_user) = 1
while (dist ≤ trust_prop_horizon) do
dist + +
foreach u in users[dist]

$$trust(u) = \frac{\sum_{i=pred(u)} (trust(i) * edge(i,u))}{\sum_{i=pred(u)} (trust(i))}$$

Figure 2. Pseudo code for the MoleTrust Algorithm

AT-MRFTrust

In contrast to MoleTrust we used and expanded MRFTrust developed in [23]. MRFTrust is designed as a belief propagation algorithm for MRF's and overcomes the coverage limitations in MoleTrust. MRFTrust constructs a network in much the same way as MoleTrust but uses undirected links. In addition, each node may be attached to evidence nodes. An evidence node represents a respective user's observation about the target node. The evidence node, thus, creates an indirect link between two nodes, For example, in Figure 3, if both Bob and Dave rated one or more articles written by Mark, they have created indirect links based on this evidence about Mark. When neither Bob,

nor Dave has direct links to Mark, a prediction can still be made through these indirect links.

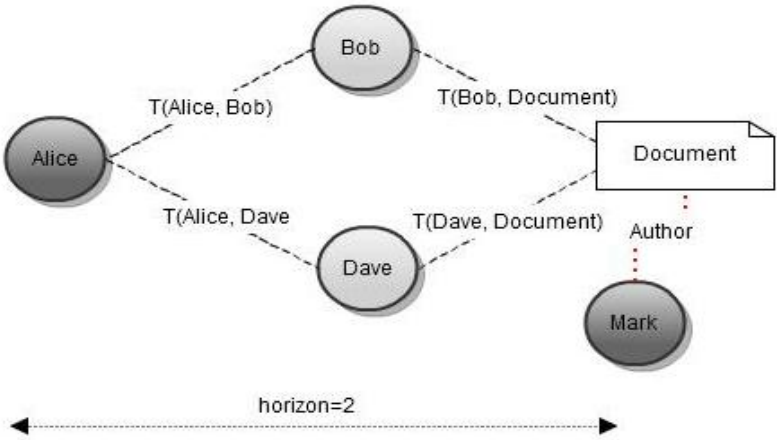

Figure 3 Determining Coverage in MRFTrust with Evidence

We modified the MRFTrust algorithm considerably to create the anytime version we call AT-MRFTrust. We also modified the MoleTrust algorithm to behave in an anytime fashion, though we did not implement the entire set of changes we made to the MRFTrust since there was no benefit and the modifications were not part of the comparison. Specifically, we made the MoleTrust algorithm interruptible after a fashion, in that we could signal it and it would provide what results it had gathered over the time it had been running. We did not make it re-startable, since we would not be comparing that ability with the AT-MRFTrust algorithm.

As we will see in our experiments, AT-MRFTrust exploits the article ratings in the epinions.com, and the movie ratings in the MovieLens datasets to use as evidence, to validate our experiments. From our research on the topic we found no other algorithm

thus far that uses other sources of evidence for trust propagation. Unlike MoleTrust, AT-MRFTrust propagates messages from the neighbors of the target node. This is based on the assumption that the neighbors of the target node have a better estimate of trust. A user can be globally more controversial even though they may be less controversial in the local graph. Thus, propagating from neighbors to the source node is in essence propagating a more reliable assessment to the source resulting in a better prediction. When constructing the graph, we ensure that the resulting graph is a tree to reduce the computational complexity of the belief propagation algorithm. For example, Figure 4 shows the result of generating the tree from the graph constructed in Figure 3.

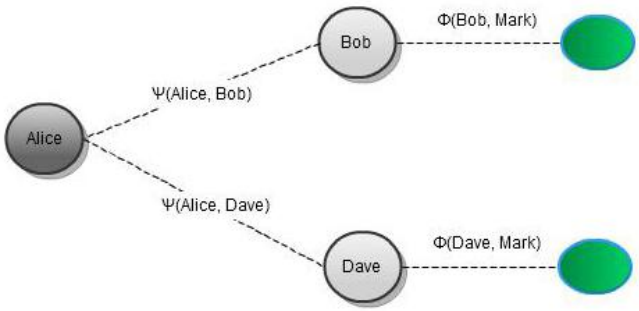

Figure 4. MRFTrust Network with Evidence nodes

The marginal at the source node represents the probability that the person associated with the source node will trust the person associated with the target node. The process by which the subgraph/tree is constructed results in that marginal being conditioned on the evidence about the target node.

For example, if Bob rated three articles written by Mark with an average score of 4.0 out of maximum score of 5.0, the local evidence is $[0.8\ 0.2]^T$. Thus with a probability

of 0.8, Bob trusts Mark. Equivalently there is a probability of 0.2 that Bob distrusts Mark. Since the inference needs to be done on-line, the complexity of the inference needs to be kept linear, consequently the propagation in MRFTrust is done in only one direction. Thus, the marginal probability distribution is only approximate. The compatibility matrix is then constructed as:

$$\psi(x_k, x_j) = \begin{bmatrix} T(x_k, x_j) & 1 - T(x_k, x_j) \\ 1 - T(x_k, x_j) & T(x_k, x_j) \end{bmatrix}$$

$T(x_k, x_j)$ represents the average value of trust between the nodes *k* and *j*. For example, if node *k* trusts node *j*, the value is 1.0. A value of 1.0 effectively means that users are compatible. On the other hand, if node *k* trusts *j*, but *j* does not trust *k*, its value is 0.5. In MRFTrust the prediction is made if at least 10 users have evidence on the target user or the target user is in the local network. Otherwise, a prediction is not made. Finally, to reduce the computational complexity for larger horizons we re-implemented both MRFTrust and MoleTrust, so that we could effectively sample the neighbors according to a threshold value. Specifically, the algorithms would only look at a sample of the neighbors to use for trust propagation. We left all settings of both MRFTrust* and MoleTrust* the same. The * indicates a wildcard that is a place holder for the horizon value (e.g,, MRFTrust2 indicates the local network is constructed with a horizon value of 2). Figure 5 shows the pseudo code for the AT-MRFTrust algorithm.

Step 1:

Input: source_user, trust_net, trust_prop_horizon, sampling_method, sampling_threshold
dist = 0;
users[dist] = source_user
if(evidence) add evidence node
while (dist ≤ trust_prop_horizon) do
dist ++
users[dist]=users reachable from users[dist - 1] and not yet visited
select edges from users[dist - 1] to users[dist] using sampling_method
keep only sampling_threshold edges

Step 2:

Output: trust scores for users
dist = 0;
trust(source_user) = 1
while (dist ≤ trust_prop_horizon) do
dist ++
foreach u in users[dist]:
propagate trust:

$$trust(u) = \sum evid(u)\ pred(u) \prod_{k \in Neighbors(u)} trust(k)$$

Figure 5. AT-MRFTrust pseudo code

eBay

The EBay algorithm is a global trust algorithm. We have implemented the eBay trust algorithm similarly to MoleTrust. Let us now specify the global trust metric we used in our experiments. The choice was guided by the fact that statements are binary (1 and 0) and not continuous. So, we chose to use a very simple metric that is similar to the one used by the online auctions site eBay.com. In order to predict the trust score of one user, the global trust metric (in the following called eBay) simply computes the fraction of received trust statements over all the received statements. Formally,

$$trust_{eBay}(user) = \frac{\#trust}{\#trust + \#distrust}$$

The trust score of a user with 0 received statements is not predictable. We considered using more complex global metrics but none of them seemed suited for the available data.

In particular, PageRank[14], probably the most advanced global metric, is not suited for the task since the original formulation does not take into account the concept of negative links and also does not produce an “authority" value in the interval [0, 1]; adapting it would have meant changing it profoundly and introducing additional biases and noises. It should be noted that, even if the input trust statement values are just 1 and 0, both the local and global metric predict trust scores in the interval [0, 1].

Sampling

As our horizon values increased the amount of runtime increased greatly. The larger the local networks that were created, the longer the runtimes needed to analyze and search those networks. Arbitrarily increasing the horizon value was insufficient to our needs as it quickly became too ponderous even for the equipment we had available. We needed to come up with a method for decreasing the runtime, yet maintaining accuracy. We also wanted to maintain a high degree of confidence in our results as well. This led to various sampling techniques that we used to manage the size of the local networks that we created. We initially began by using a static threshold for the number of random neighbors that were included for each node. The number of neighbors for each user varies greatly in each dataset so using a static number of neighbors seemed to be an obvious way of ensuring fair inclusion of neighboring nodes. The difficulty with this method quickly became obvious as the results were extremely poor as seen in Figures 6 and 7.

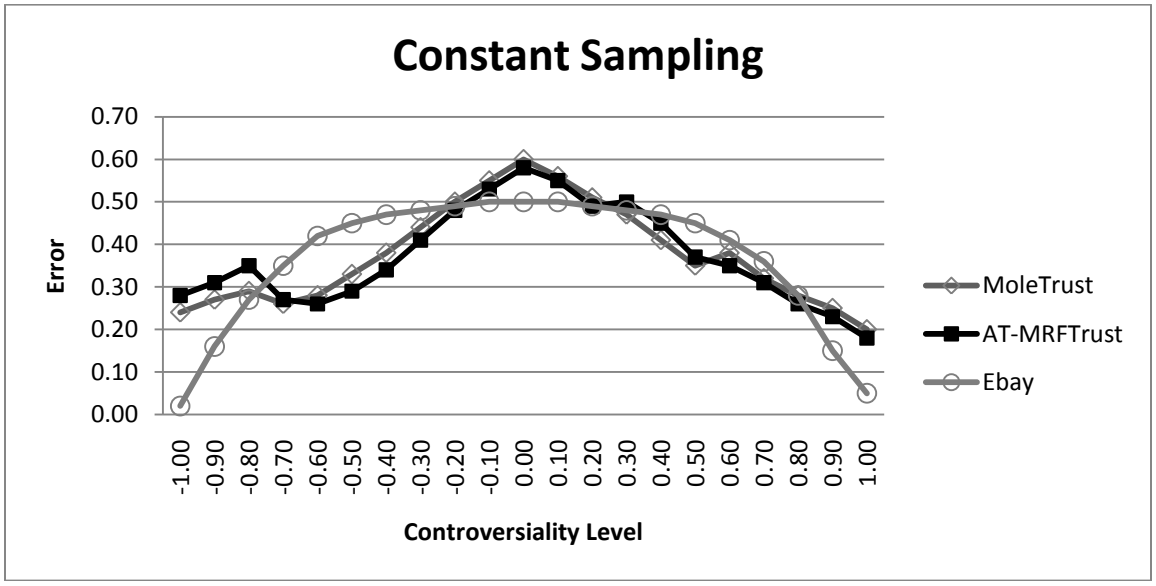

Figure 6 Constant Sampling using 5 neighbors

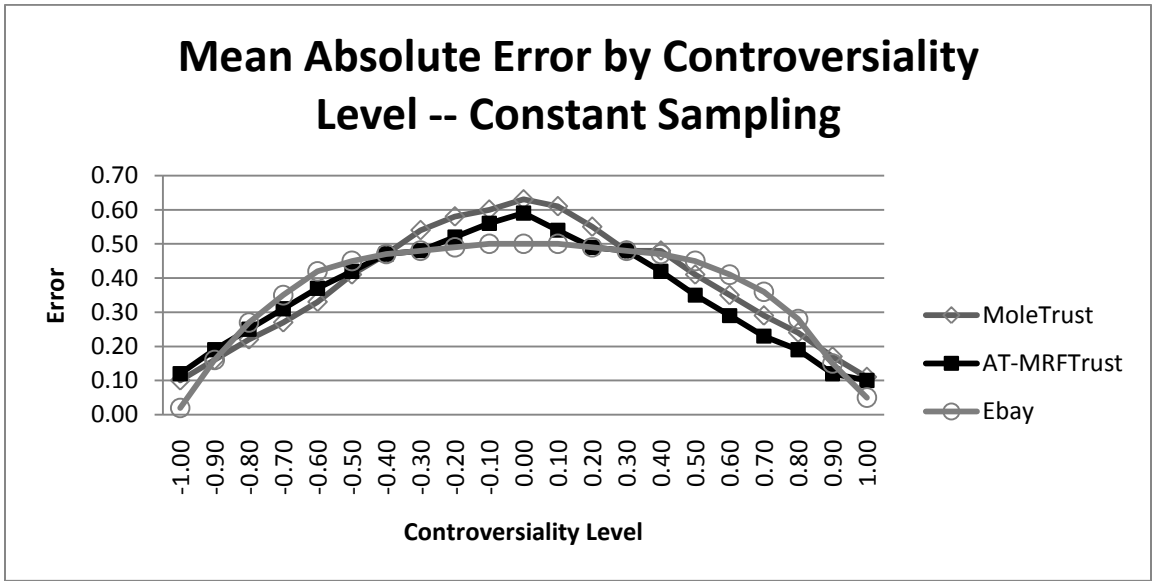

Figure 7 Constant Sampling Accuracy using 10 neighbors

We attempted various static settings with little improvement. Next, we tried sorting the neighbors by trust statement and picking a static number from both ends of the spectrum. This also produced results that were very poor, and the error levels were very high. Our next attempt was instead of using a static threshold, to use a variable threshold

based on the number of neighbors and to choose those neighbors randomly. Our first threshold was 50% and the results from these experiments were very promising as the error levels were low as we show in Figure 8.

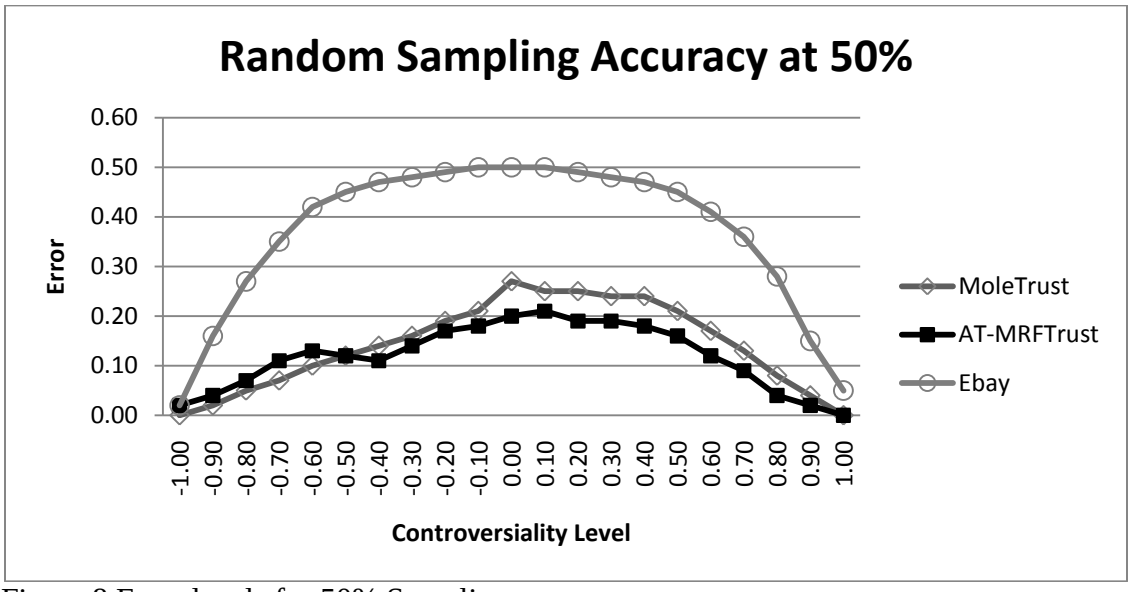

Figure 8 Error levels for 50% Sampling.

At this point our task became finding the level at which the error levels were comparable to error levels of the full network while maintaining high confidence levels. Our next attempt was at 25% and while those experiments gave an excellent reduction in the runtime, the confidence levels dropped to unacceptable levels as shown in Figure 9.

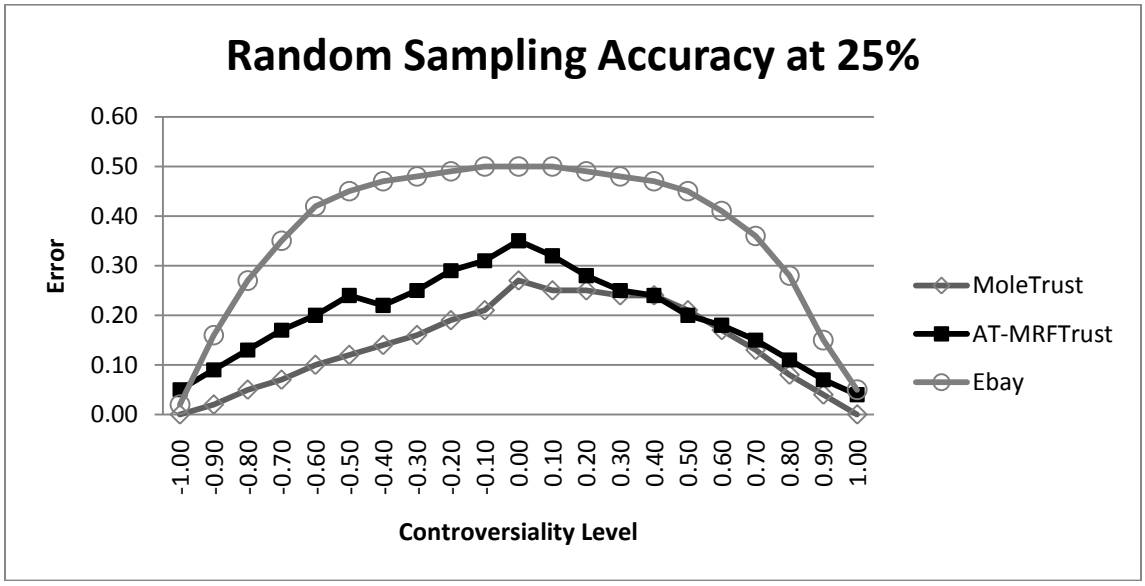

Figure 9 Sampling Accuracy at 25%

In our various experiments we settled on 35% as the level where we got acceptable results as we will demonstrate later. In Figure 10 we see the results of comparing AT-MRFTrust using sampling at 50% and without sampling. AT-MRFTrust-S in the figure is the version that used sampling. This demonstration of the accuracy convinced us that we were on the correct path. The experiments in the following chapter assume a 35% level and all configurations use this level of sampling.

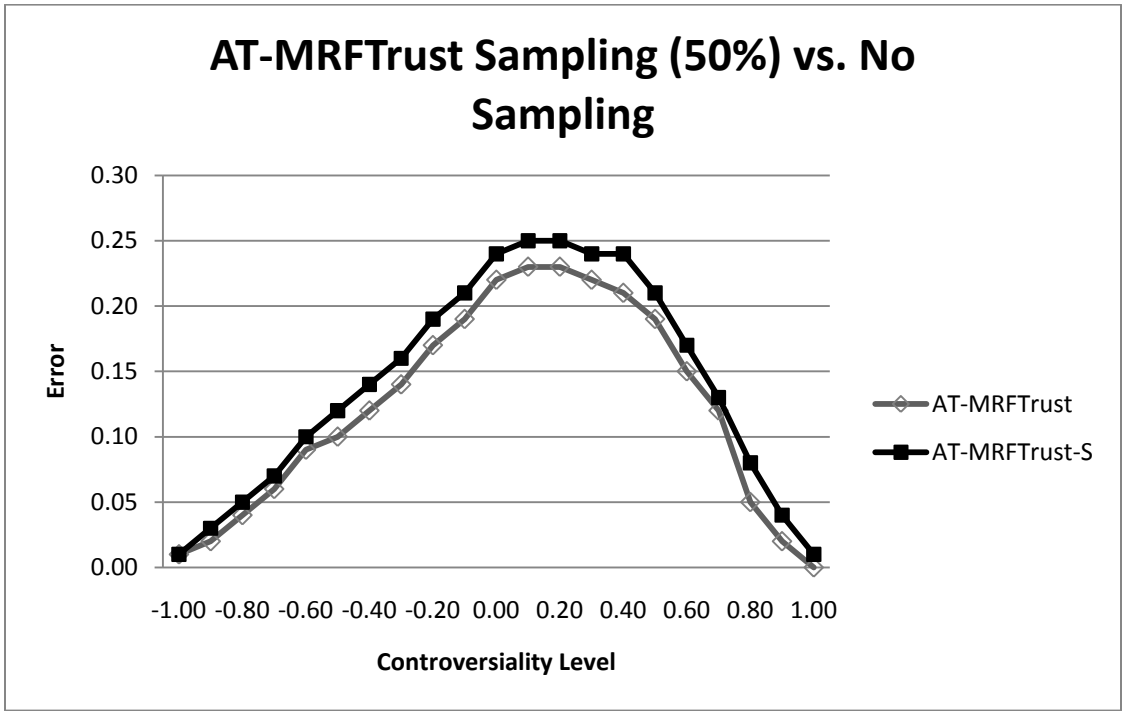

Figure 10 AT-MRFTrust with and without sampling turned on at Time Step 2.

To create AT-MRFTrust, we combined our sampling technique with an increase in horizon value as the algorithm runtime increased. We also ensured that after the initial burn-in time that the nodes always had a trust value. This allowed us to make the algorithm interruptible. Once the algorithm was made interruptible and re-startable, we could then re-initialize the algorithm with the results of a previous run, thereby saving substantial processing time. Figure 11 shows the various error levels for the sampling techniques used to determine where we felt the tradeoff was best for performance versus accuracy.

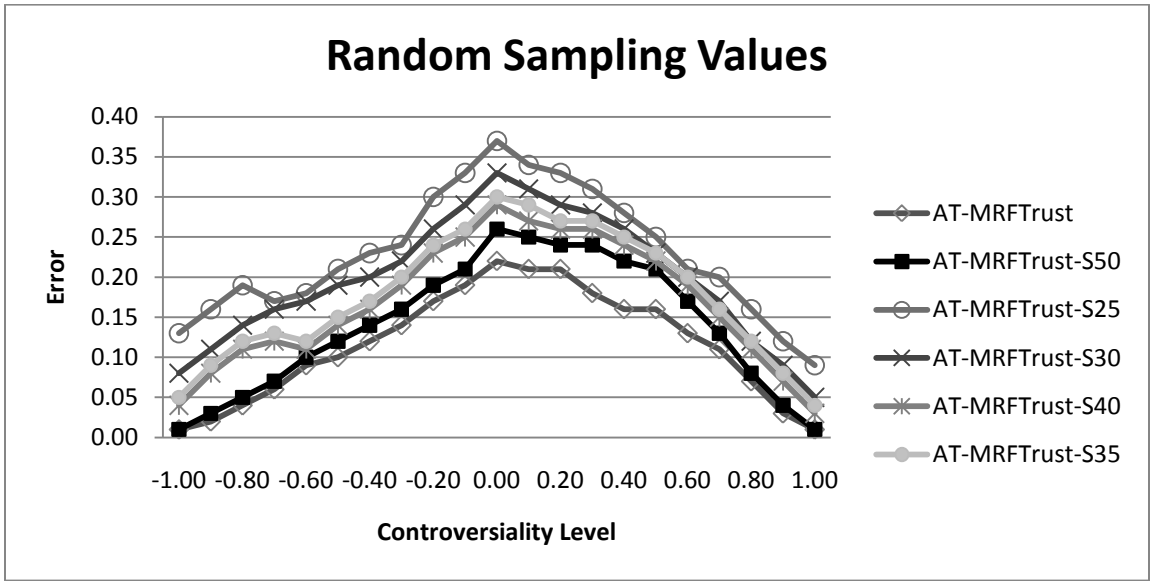

Figure 11 Random Sampling levels. The numbers at the end indicate sampling percentage.

EXPERIMENTS AND ANALYSIS

In this study we tested our algorithm on two different datasets. The Epinions.com dataset[25] and the MovieLens[26] movie recommender dataset.

Epinions.com is a website where users can write reviews about products and rate other reviews. It also allows users to express their Web of Trust (i.e., reviewers they trust) and their Block List (i.e., a list of authors they distrust). Thus, the dataset contains user ratings of articles created by other users as well as user ratings of each other in the form of the Web of Trust and the Block List. The value of 1 is used for trust statements (Web of Trust), and -1 is used for distrust statements (Block List). We rescaled the trust scores so that the distrust statements would have a value of 0. This ensured the trust values were in the range 0.0…1.0. The article ratings represent how likely a user rates a certain textual article written by another user. The rating value has a range 1…5 where 1 is "not helpful" and 5 is "very helpful." The epinions.com data set contains about 132,000 users, who issued 841,372 statements of trust or distrust. We removed 573 trust statements corresponding to statements where the recipients of the statements were also the senders of the same statements. In testing our approach, we used a leave-one-out experimental design to evaluate the performance of the model. Specifically, for every existing relationship between users A and B in the data set, we removed a true trust statement from user A to user B and then constructed the local network between them. Then, we predicted A's trust statement for B based on that network.

Our other dataset was the MovieLens online recommender service. It had 10,000,054 ratings of 10,681 movies from 71,567 users. The users in the dataset were

included only if they had rated 20 or more movies. The ratings system for MovieLens allows users to rate using a star system. The star ratings range from half stars to 5 stars in increments of 0.5. So that it would match up with our implementation we have rounded all ratings up to the nearest integer. This allowed the rating value to go from 1…5 where 1 is "not good" and 5 is "very good". We then used the trust statements as in the epinions.com dataset to create our Web of Trust for this dataset. We, again, used a leave-one-out cross-validation scheme to evaluate performance. The movies were converted to "user" entities to give the same structure to the data as the epinion.com data. This prevented us from having to rewrite the code entirely and gave comparable results.

To provide sufficient and fair analysis we compared three algorithms in our study: our algorithm, AT-MRFTrust, our modified version of MoleTrust, and the eBay algorithm.

Our experiments demonstrate that the AT-MRFTrust algorithm performs better as a function of runtime and reduces error levels as the algorithm is allowed to run. It performs better than both the original MRFTrust algorithm and the MoleTrust algorithm. Figure 12 shows the mean absolute error after initial burn-in. Obviously, AT-MRFTrust does not perform as well as MoleTrust after the burn-in period, but performs significantly better than the global trust score of the eBay algorithm. This comparison is a bit unfair as the MoleTrust algorithm does not produce results at the same time step as the AT-MRFTrust, good, bad, or otherwise. We use it here to demonstrate that the burn-in accuracy is still reasonable and the runtime is approximately half an hour for the

algorithm to get to this point, compared to about 6 hours for MoleTrust to get to the same point.

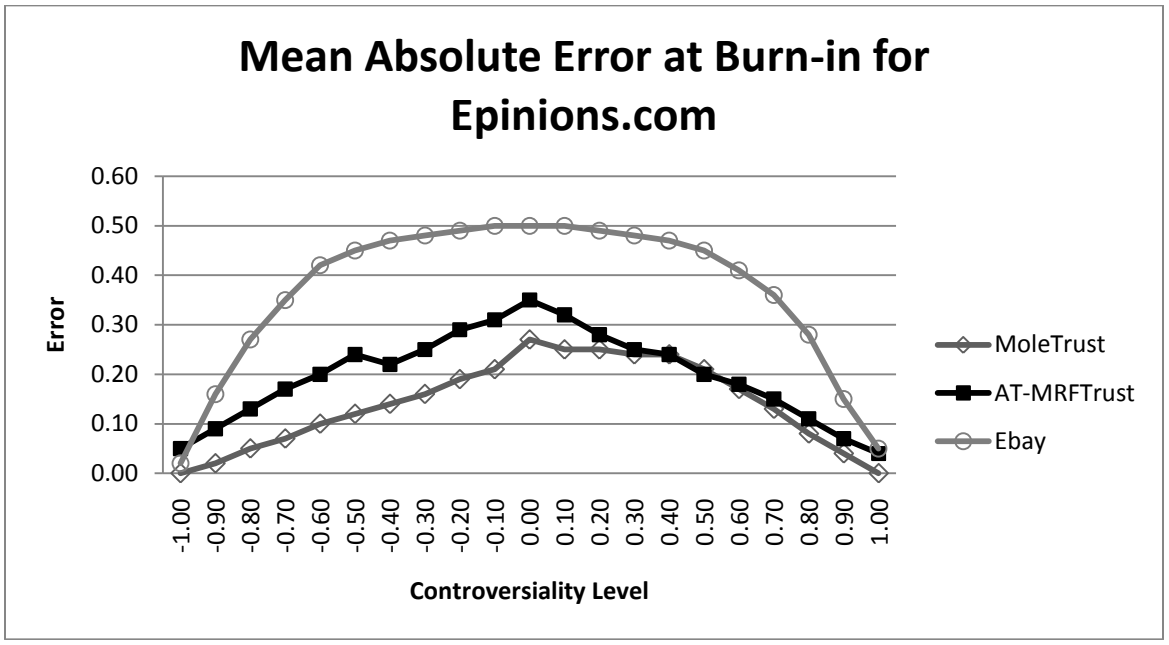

Figure 12 Performance after Burn-in for epinions.com data

Figure 13 shows the Mean Absolute Error for the MoleTrust, AT-MRFTrust, and eBay algorithms at time step 3. Figure 14 shows the Mean Absolute Error for the three algorithms at time step 3 for the MovieLens data set. Figures 15 and 16 shows the Error rate at time step 4 for Epinions.com and MovieLens, respectively. Time steps are measured by a complete propagation of the algorithm through the users. As demonstrated in the figure above, the eBay global trust algorithm is very poor and wrong half the time when predicting the trust for the most controversial users. MoleTrust outperforms AT-MRFTrust in the most distrusted users ranging between -1.0 and -0.5 for

horizon 1. For the most controversial and the most trusted users, AT-MRFTrust outperforms MoleTrust.

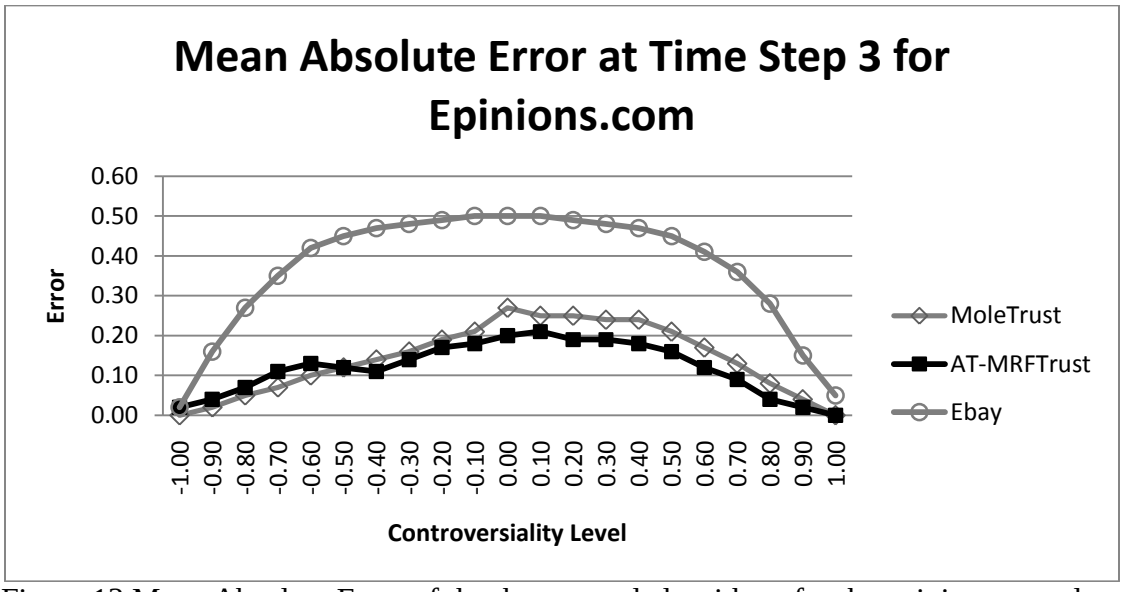

Figure 13 Mean Absolute Error of the three tested algorithms for the epinions.com data at time step 3.

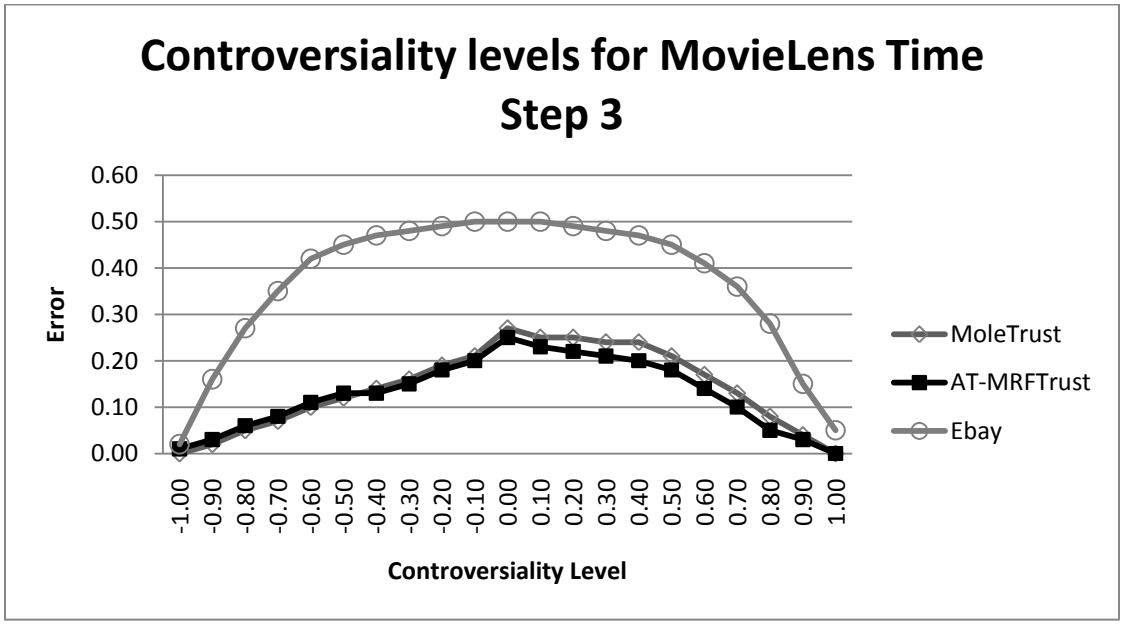

Figure 14 Mean Absolute Error of the three tested algorithms at time step 3 for MovieLens

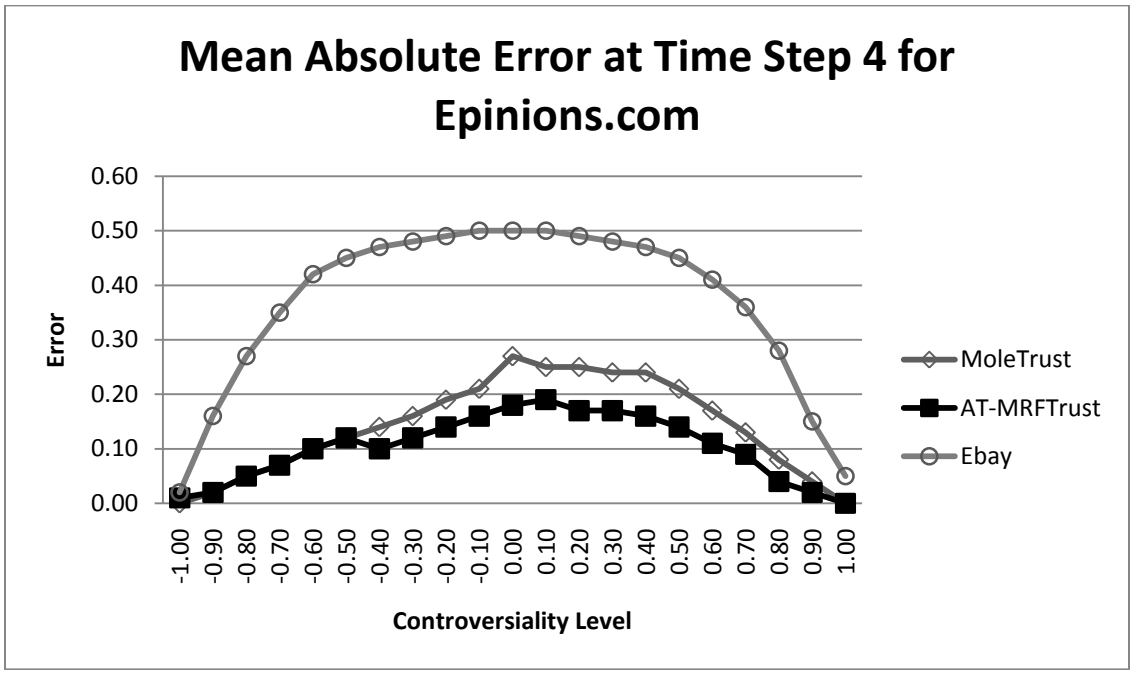

Figure 15 Mean Absolute Error for epinions.com at time step 4

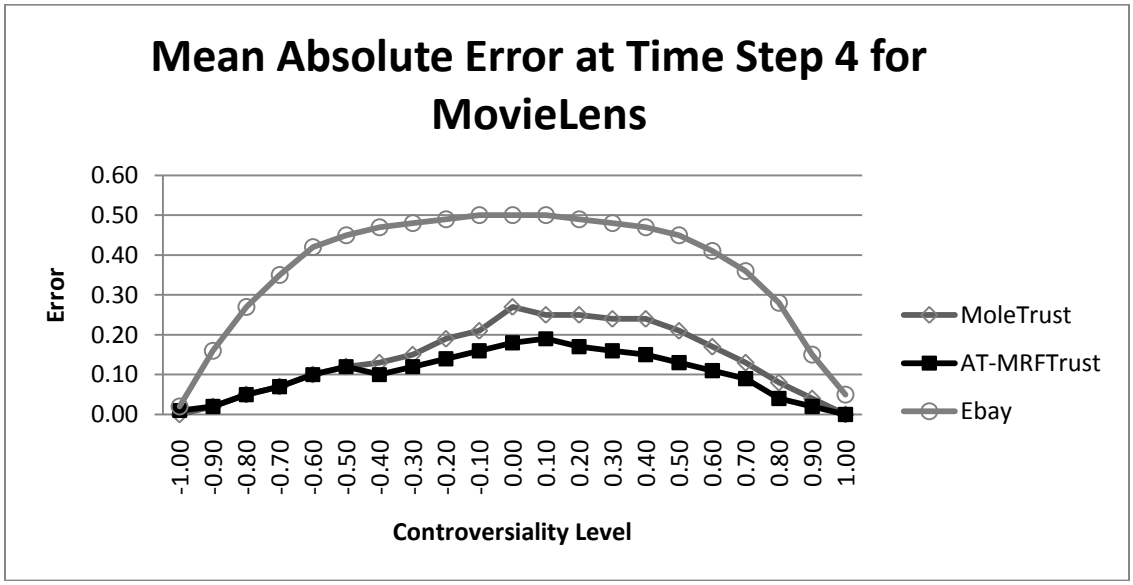

Figure 16 Mean Absolute Error for MovieLens at Time Step 4

The increase in runtime reduces the error levels despite the sampling technique used. Figure 17 also shows the improvement of the AT-MRFTrust algorithm as runtime

increases for the Epinions.com data set. By time step 5, which is approximately 10 hours, the AT-MRFTrust outpaces the MoleTrust algorithm. The eBay algorithm does not change. The larger the horizon and the more nodes included with network the more accurate the result is intuitive. Figure 18 demonstrates the same improvement in the MovieLens data set.

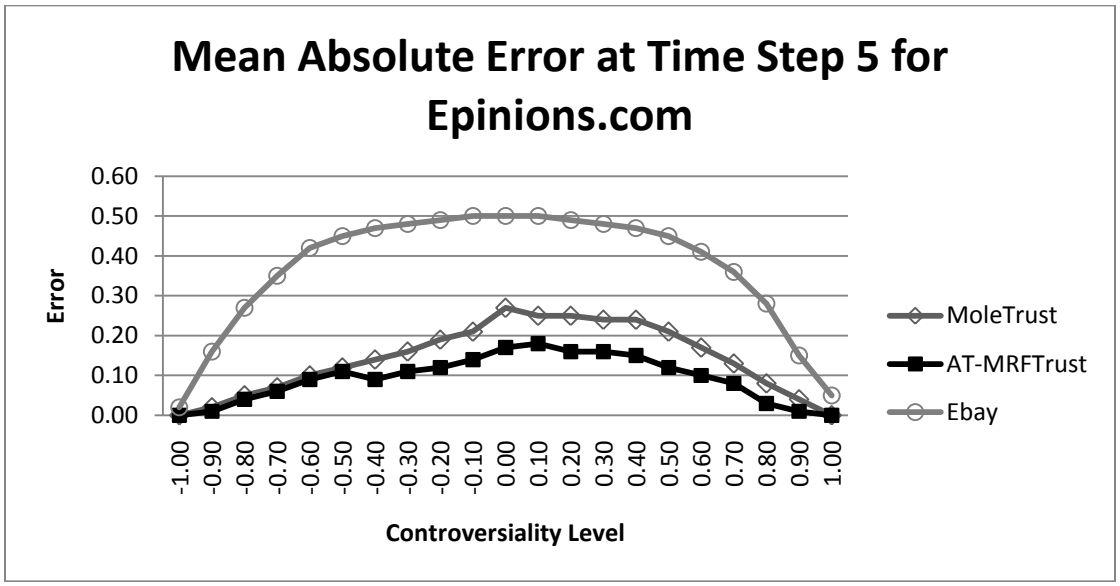

Figure 17 Mean Absolute Error for epinions.com at time step 5

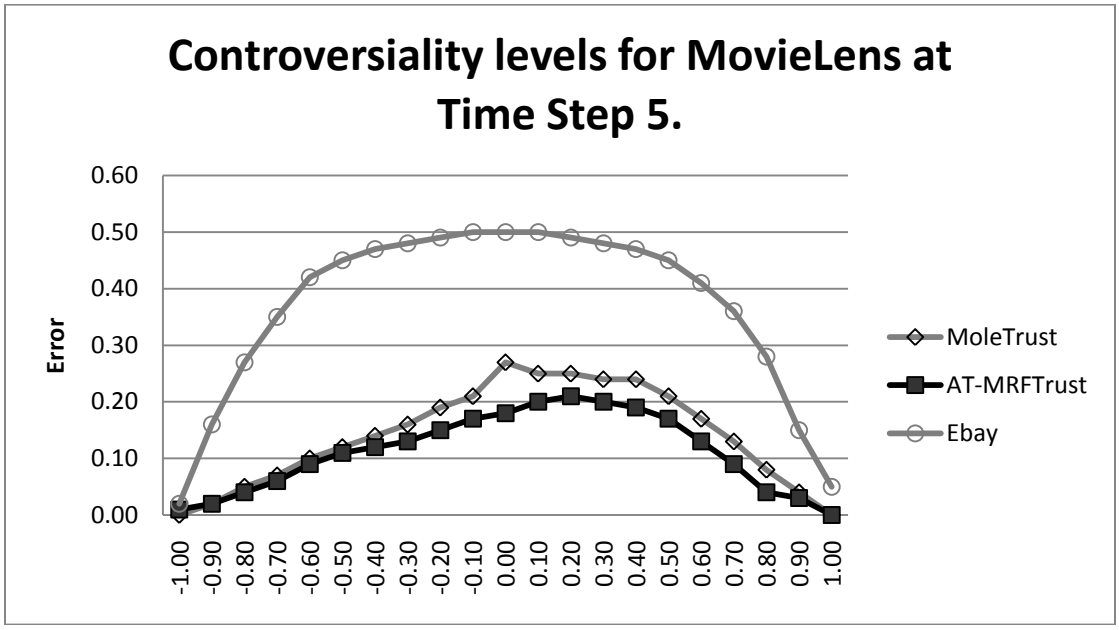

Figure 18 Controversiality Levels for the three tested Algorithms at Time Step 5 for MovieLens.

Due to the threshold used for the MoleTrust algorithm for propagation, along with the fact that the target user must be in the local network, the coverage is fairly poor. As runtime increases, coverage improves, but it improves slowly in the case of MoleTrust. In Figure 19 we demonstrate the increase in coverage as compute time increases. The 2 and 3 in the graph are step indicators that coincide with time step increases. Coverage is very good in the AT-MRFTrust algorithm due to the inclusion of evidence. MoleTrust coverage increases slowly since a prediction can only be made if the target is included in the local network and the trust threshold value of 0.6 keeps the propagation from improving significantly. Figure 20 shows the results of using the MoleTrust algorithm with and without the threshold. MoleTrust considers users with a trust value of less than that to be untrustworthy and their trust statements are discarded. This is a fundamental constraint of the MoleTrust algorithm and removing it distorts the algorithm and the error

is very high, especially in the area of distrust, however the numbers still are better than flipping a coin.

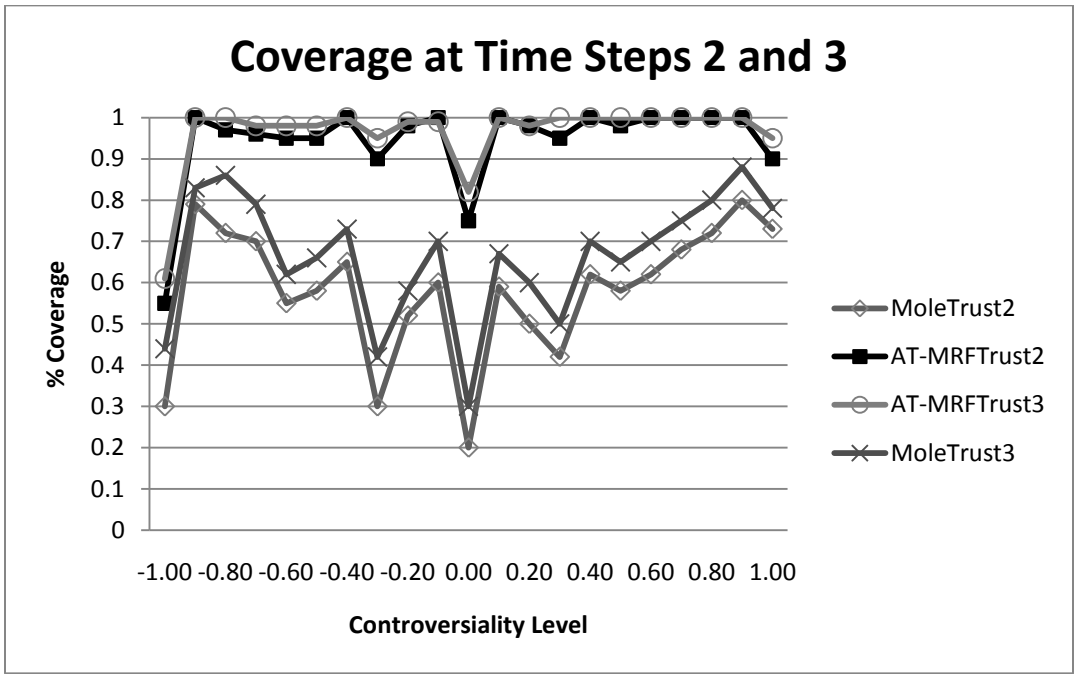

Figure 19 Coverage at time step 2 & 3 for AT-MRFTrust and MoleTrust.

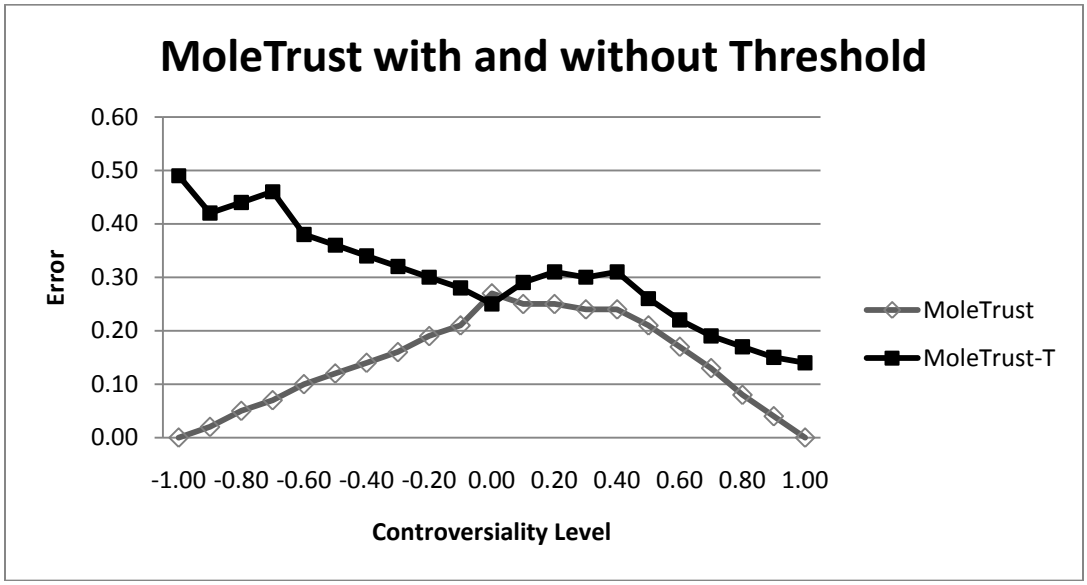

Figure 20 MoleTrust-T has the threshold removed, while the other uses the threshold

Table 1 shows the 95% confidence intervals at time step 2 for the epinions.com. Early in the process the confidence intervals for MoleTrust are better than for MRFTrust and AT-MRFTrust. This is expected since both the MRFTrust and AT-MRFTrust algorithms make predictions where MoleTrust does not. Table 2 shows the same intervals over the MovieLens dataset for time step 2. The values for AT-MRFTrust and the original MRFTrust are very similar as expected, but not identical considering the random nature of choosing the neighbors to include in the local network. The significance of the difference can be seen by looking at the table. For MoleTrust at the -1.0 controversiality level the variance is .0008 - .0020, while MRFTrust is .2493-.2593, and AT-MRFTrust is .2505-.2611. The difference is quite significant in the case of MoleTrust versus the other two algorithms at the distrust levels. Looking further into the table, as the levels become less controversial, the difference takes on less significance. For example at the controversiality level of 0.6 the variance for MoleTrust is .2212-.2304, while that of MRFTrust is .2162-.2260 and AT-MRFTrust .2180-.2287. The fact that these numbers overlap by this point means that they have become insignificant.

	epinions.com		
Cntr	MoleTrust	MRFTrust	AT-MRFTrust
-1.0	0.0014±0.0006	0.2543±0.0050	0.2558±0.0053
-0.9	0.0270±0.0037	0.1787±0.0072	0.1857±0.0084
-0.8	0.0569±0.0060	0.2263±0.0094	0.2333±0.0104
-0.7	0.0714±0.0071	0.2180±0.0102	0.2200±0.0112
-0.6	0.1052±0.0129	0.2902±0.0143	0.2974±0.0161
-0.5	0.1231±0.0114	0.2838±0.0126	0.2881±0.0144
-0.4	0.1723±0.0139	0.3056±0.0158	0.3106±0.0167
-0.3	0.1696±0.0170	0.3326±0.0132	0.3351±0.0143
-0.2	0.1855±0.0163	0.3053±0.0155	0.3090±0.0171
-0.1	0.2151±0.0136	0.2958±0.0143	0.2975±0.0159
0.0	0.2904±0.0189	0.3322±0.0101	0.3349±0.0118
0.1	0.2409±0.0109	0.3138±0.0116	0.3162±0.0129
0.2	0.2479±0.0126	0.2976±0.0110	0.2996±0.0113
0.3	0.2466±0.0100	0.2866±0.0085	0.2836±0.0099
0.4	0.2588±0.0077	0.2513±0.0084	0.2533±0.0088
0.5	0.2422±0.0066	0.2456±0.0066	0.2471±0.0073
0.6	0.2258±0.0046	0.2211±0.0049	0.2232±0.0055
0.7	0.1692±0.0035	0.1724±0.0038	0.1764±0.0046
0.8	0.1183±0.0018	0.1312±0.0021	0.1331±0.0030
0.9	0.0555±0.0008	0.0792±0.0010	0.0810±0.0013
1.0	0.0079±0.0003	0.0886±0.0010	0.0894±0.0011

Table 1. 95% Confidence Intervals for the Epinions.com data set at time step 2.

	MovieLens		
Cntr	MoleTrust	MRFTrust	AT-MRFTrust
-1.0	0.0012±0.0005	0.2412±0.0031	0.2468±0.0042
-0.9	0.0247±0.0032	0.1667±0.0066	0.1801±0.0077
-0.8	0.0531±0.0051	0.2193±0.0081	0.2227±0.0086
-0.7	0.0683±0.0059	0.2110±0.0090	0.2159±0.0101
-0.6	0.0972±0.0108	0.2792±0.0103	0.2893±0.0144
-0.5	0.1123±0.0097	0.2708±0.0096	0.2784±0.0127
-0.4	0.1687±0.0114	0.2906±0.0117	0.3016±0.0143
-0.3	0.1656±0.0131	0.2826±0.0101	0.3151±0.0152
-0.2	0.1799±0.0123	0.2931±0.0115	0.3063±0.0138
-0.1	0.2048±0.0108	0.2844±0.0109	0.2870±0.0120
0.0	0.2794±0.0136	0.3215±0.0088	0.3247±0.0102
0.1	0.2345±0.0089	0.3029±0.0102	0.3091±0.0129
0.2	0.2402±0.0115	0.2800±0.0093	0.2996±0.0099
0.3	0.2397±0.0072	0.2771±0.0084	0.2798±0.0087
0.4	0.2501±0.0059	0.2403±0.0078	0.2429±0.0081
0.5	0.2362±0.0048	0.2342±0.0047	0.2383±0.0063
0.6	0.2138±0.0031	0.2121±0.0036	0.2152±0.0045
0.7	0.1573±0.0023	0.1614±0.0029	0.1665±0.0032
0.8	0.1064±0.0018	0.1202±0.0016	0.1247±0.0021
0.9	0.0465±0.0007	0.0697±0.0008	0.0740±0.0011
1.0	0.0049±0.0002	0.0816±0.0007	0.0856±0.0008

Table 2 95% Confidence intervals at time step 2 for the MovieLens Data.

As we see in Tables 3 and 4, the AT-MRFTrust algorithm has improved past that of MoleTrust by time step 5. This stands to reason as the networks have gotten deeper, the accuracy and reliability of the algorithm improves significantly. Early in the process MoleTrust outperforms AT-MRFTrust in accuracy, especially in the most distrusted area. The MRFTrust algorithm has no data by time step 5 so it is not included in the table. In analyzing the tables, looking at the -1.0 controversiality level, for MoleTrust the error interval is .0007 to .0017 and for AT-MRFTrust they are .0006 to .0020. The fact that they overlap shows that the differences have become insignificant in this range by time

step 5. We cannot reject the null hypothesis that the data comes from the same distribution.

	epinions.com	
Cntr	MoleTrust	AT-MRFTrust
-1.0	0.0012±0.0005	0.0013±0.0007
-0.9	0.0250±0.0034	0.0230±0.0030
-0.8	0.0552±0.0056	0.0541±0.0051
-0.7	0.0691±0.0066	0.0659±0.0058
-0.6	0.0977±0.0122	0.0961±0.0103
-0.5	0.1198±0.0107	0.1132±0.0089
-0.4	0.1701±0.0131	0.1556±0.0111
-0.3	0.1683±0.0162	0.1597±0.0132
-0.2	0.1826±0.0155	0.1708±0.0126
-0.1	0.2139±0.0129	0.2001±0.0110
0.0	0.2890±0.0177	0.2583±0.0128
0.1	0.2394±0.0101	0.2138±0.0091
0.2	0.2459±0.0120	0.2219±0.0108
0.3	0.2448±0.0093	0.2167±0.0075
0.4	0.2576±0.0068	0.1880±0.0071
0.5	0.2412±0.0059	0.1765±0.0059
0.6	0.2249±0.0039	0.1531±0.0040
0.7	0.1681±0.0027	0.1223±0.0031
0.8	0.1170±0.0013	0.0818±0.0017
0.9	0.0543±0.0007	0.0322±0.0005
1.0	0.0071±0.0006	0.0046±0.0004

Table 3 95% Confidence Intervals at Time Step 5 for Epinions.com

	MovieLens	
Cntr	MoleTrust	AT-MRFTrust
-1.0	0.0011±0.0005	0.0034±0.0009
-0.9	0.0239±0.0030	0.0228±0.0029
-0.8	0.0523±0.0047	0.0500±0.0044
-0.7	0.0657±0.0054	0.0633±0.0070
-0.6	0.0948±0.0101	0.0897±0.0091
-0.5	0.1106±0.0093	0.1003±0.0083
-0.4	0.1655±0.0109	0.1422±0.0096
-0.3	0.1632±0.0122	0.1551±0.0107
-0.2	0.1754±0.0116	0.1607±0.0105
-0.1	0.2019±0.0099	0.1909±0.0088
0.0	0.2794±0.0136	0.2395±0.0121
0.1	0.2345±0.0089	0.2128±0.0083
0.2	0.2402±0.0115	0.2203±0.0101
0.3	0.2397±0.0072	0.2132±0.0070
0.4	0.2501±0.0059	0.1837±0.0068
0.5	0.2362±0.0048	0.1715±0.0051
0.6	0.2138±0.0031	0.1496±0.0030
0.7	0.1573±0.0023	0.1177±0.0017
0.8	0.1064±0.0018	0.0768±0.0012
0.9	0.0465±0.0007	0.0299±0.0004
1.0	0.0049±0.0002	0.0039±0.0003

Table 4 95% Confidence Intervals for MovieLens Dataset at time step 5.

Figures 21 and 22 demonstrate the mean absolute error over time. These are averaged over 12 runs of the data and averaged over the controversiality levels. As can be seen from the figures, the AT-MRFTrust algorithm improves with additional runtimes. The MoleTrust algorithm improves slightly over time, but not statistically significantly. The eBay algorithm, since it is a global trust value, does not improve over time and its initial value is its maintained value. Figure 21 shows results on the epinions.com dataset while Figure 22 shows results on the MovieLens dataset. The differences in the datasets are slight and the results are nearly mirror images. The MovieLens data is slightly more

accurate since it has some built-in filters to ensure that the data was more statistically credible. These filters ensure that a user has rated at least 20 movies before they were included in the data set. There are considerably more trust statements in the MovieLens data so it took significantly longer to reach the same level of "maturity" as the epinions.com data. It did not, however, take as much longer as we expected, thus demonstrating that the algorithm scales fairly well. We were expecting more linear runtime with the increase in data size.

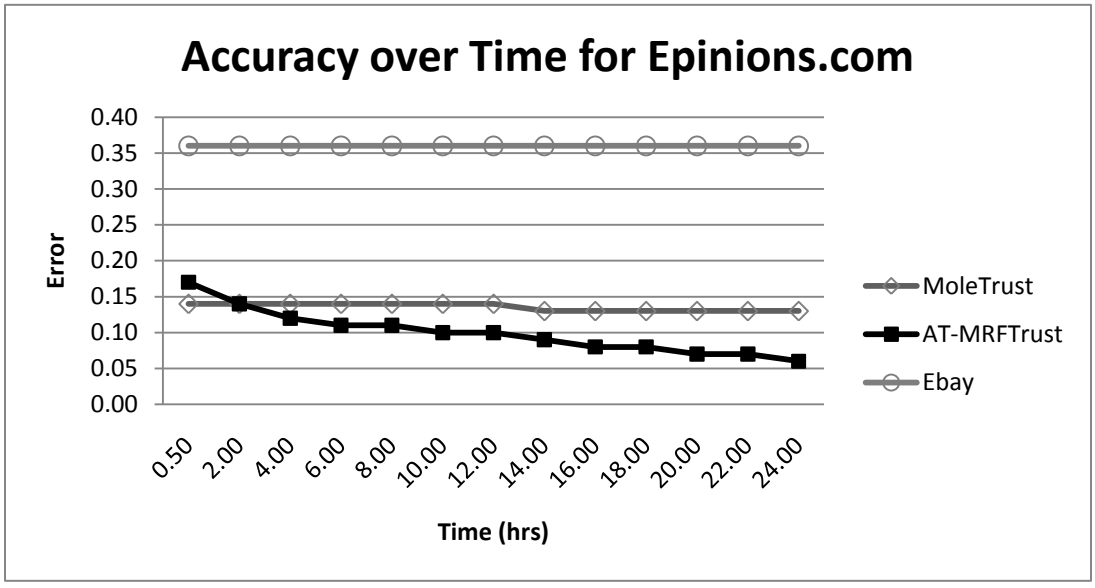

Figure 21 Mean Absolute Error over time for epinions.com

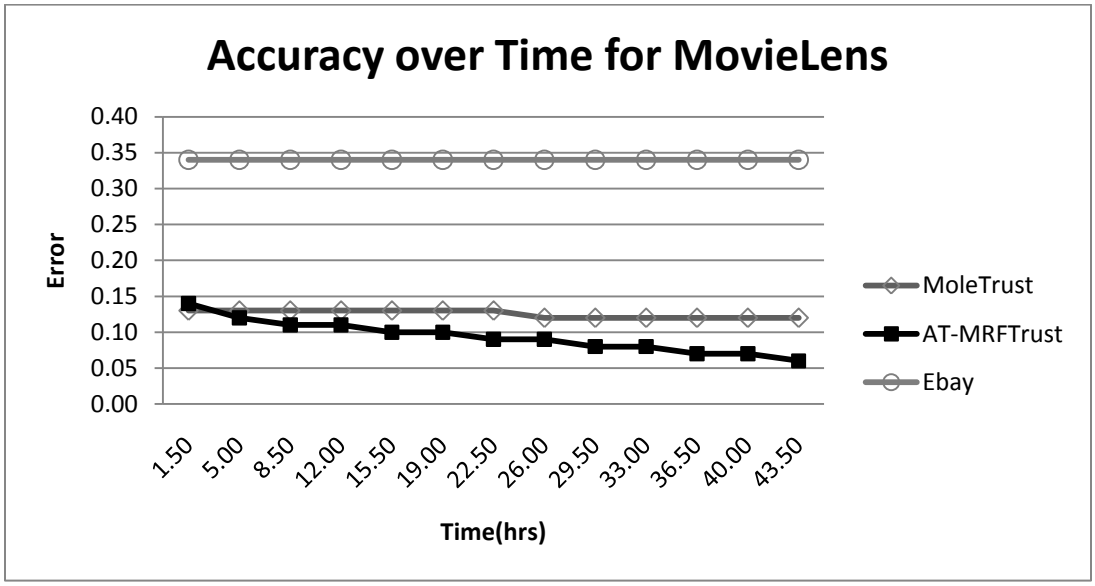

Figure 22 Mean Absolute Error for MovieLens data

In order to perform longer runs of the data we developed a number of automated tools to keep the data running in various formats and complete post-processing with minimal delays. We used 4 Dell R710's with 16 processing threads on each machine to divide up the work load. This gave us 64 processing threads to work with. Each controversiality interval was processed on a single thread and the monitor processes watched the processing flows to keep the data processing, each complete run consuming 21 of the processing threads. We were able to run 3 different experiments simultaneously in this hardware set up. We were able to run nearly continuously for more than 2 months. Figure 23 shows the runtimes of various runs. These are averages over a number of runs. As seen in the chart, the runtimes for 25-40 percent sampling rates are very close in time, relatively speaking. Since the graph we use is actually converted to a tree the worst case runtime for the algorithm is $C(|V|\times|E|)$. In our reductions we only reduce the size of the constant. Our bound on the runtime is then considered to be $O(|V|\times|E|)$.

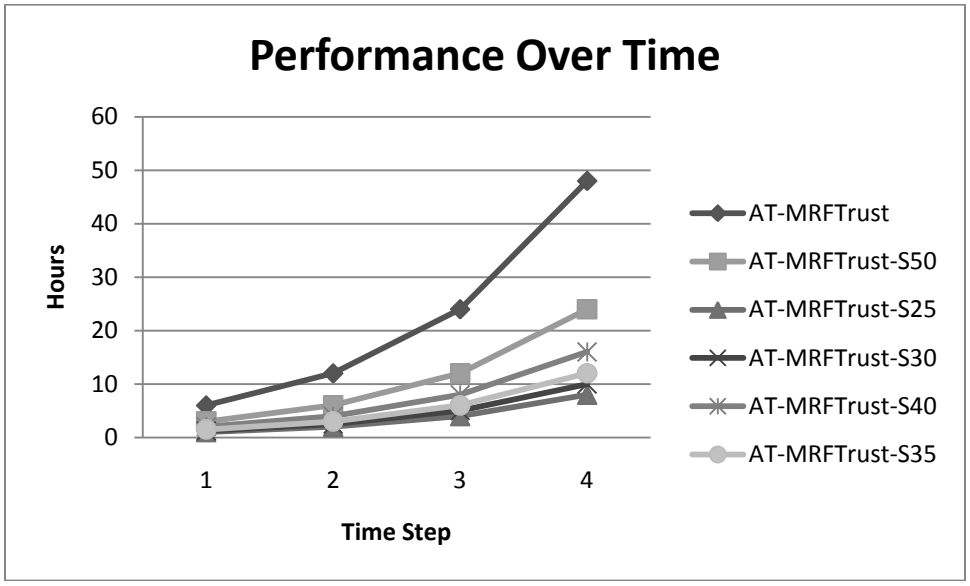

Figure 23 Performance over Time

CONCLUSIONS AND FUTURE WORK

We have proposed an anytime algorithm to perform trust propagation on social networks. The use of the ratings of both articles in the epinions.com and movies in the MovieLens data sets allowed us to test the validity of the algorithm. Our algorithm uses Markov Random Fields and incorporates evidence to leverage information that exists in the data to produce more accurate results. We have implemented a number of techniques to improve runtimes and accuracies, including sampling techniques to reduce the size of the local networks. We had to overcome a number of complexities to ensure a fair test. We had to modify the MoleTrust algorithm considerably to perform equitable tests. Coverage between the two local trust algorithms was significantly different due to the threshold that defines the MoleTrust algorithm and the trust propagation of that model. We did implement some successful algorithm modifications to equalize the coverage issues and lend more credibility to the results we obtained. These results have been presented in previous works and are not covered here. The biggest challenge we faced was managing the computational complexity of the network and by using some sampling techniques we were able to significantly reduce that complexity and still maintain acceptable accuracy and confidence levels.

Future work that we would like to perform is to find another local trust algorithm to test against. Using the state-of-the-art MoleTrust algorithm allowed us to understand the complexities of trust propagation and gave us something to compare to equitably. We would also like to further optimize the implementation to gain additional performance advantages. We believe that we can further improve accuracy using less runtime. We

also would like to consider using conditional random fields (CRF's) rather than MRF's to see if we can do the propagation in a single step instead of the two step method of the MRF. Another consideration would be doing bi-directional propagation. In addition, we would like to investigate creating a data filtering mechanism that would allow us to use other types of data without rewriting the code for every data set.

BIBLIOGRAPHY

[1] Judy Shapiro, "The Next Disruptive Tech on the Web? Trust", Ad Age Digital, February, 16, 2010

[2] Richardson M., Agrawal R., Domingos P. Trust Management for the Semantic Web. *Proceedings of Semantic Web Conference* 2003.

[3] C.-N. Ziegler and G. Lausen, "Spreading activation models for trust propagation," in IEEE International Conference on e-Technology, e-Commerce, and e-Service, 2004, pp. 83–97.

[4] J. Barbalet, "A characterization of trust, and its consequences," Theory and Society, vol. 38, pp. 367–382, 2009. [Online]. Available: http://dx.doi.org/10.1007/s11186-009-9087-3

[5] C. Ziegler and G. Lausen, "Analyzing correlation between trust and user similarity in online communities," in Trust Management, ser. Lecture Notes in Computer Science, C. Jensen, S. Poslad, and T. Dimitrakos, Eds. Springer Berlin / Heidelberg, 2004, vol. 2995, pp. 251–265. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-24747-019

[6] C.-N. Ziegler and G. Lausen, "Propagation models for trust and distrust in social networks," Information Systems Frontiers, vol. 7, pp. 337–358, 2005. [Online]. Available: http://dx.doi.org/10.1007/s10796-005-4807-3

[7] F. Walter, S. Battiston, and F. Schweitzer, "A model of a trust-based recommendation system on a social network," Autonomous Agents and Multi-Agent Systems, vol. 16, pp. 57–74, 2008. [Online]. Available: http://dx.doi.org/10.1007/s10458-007-9021-x

[8] J. Golbeck, "Computing and applying trust in web-based social networks. ph.d. thesis." University of Maryland, Tech.Rep., 2005.

[9] P. Massa and P. Avesani, "Controversial users demand local trust metrics: an experimental study on epinions.com community," in Proceedings of the 25th American Association for Artificial Intelligence Conference, 2005.

[10] J. Golbeck and J. Hendler, "Inferring binary trust relationships in web-based social networks," ACM Trans. Internet Technol.,vol. 6, no. 4, pp. 497–529, 2006.

[11] L. Mui, "Computational models of trust and reputation:agents, evolutionary games, and social networks. ph.d. thesis," Massachusetts Institute of Technology, Tech. Rep., 1995.

[12] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in WWW '03: Proceedings of the 12th international conference on World Wide Web. New York, NY, USA: ACM,2003, pp. 640–651.

[13] Y.-F. Wang, Y. Hori, and K. Sakurai, "Characterizing economic and social properties of trust and reputation systems in p2p environment," Journal of Computer Science and Technology, vol. 23, pp. 129–140, 2008. [Online]. Available: http://dx.doi.org/10.1007/s11390-008-9118-y

[14] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in Proceedings of the 13th international conference on World Wide Web, ser. WWW '04. New York, NY, USA: ACM, 2004, pp. 403–412. [Online]. Available: http://doi.acm.org/10.1145/988672.988727

[15] Josang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in ACSC '06: Proceedings of the 29th Australasian Computer Science Conference. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006, pp. 85–94.

[16] K. Thirunarayan, D. Althuru, C. Henson, and A. Sheth, "A local qualitative approach to referral and functional trust," in Proceedings of the The 4th Indian International Conference on Artificial Intelligence (IICAI-09), 2009.

[17] L. Ding, P. Kolari, S. Ganjugunte, T. Finin, and A. Joshi, "Modeling and evaluating trust network inference," in Proceedings of The Workshop on Deception, Fraud and Trust in Agent Societies at The Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-2004, 2004, pp. 21–32.

[18] E. Chang, E. Damiani, and T. Dillon, "Fuzzy approaches to trust management," in Computational Intelligence, Theory and Applications, B. Reusch, Ed. Springer Berlin Heidelberg, 2006, pp. 425–436. [Online]. Available: http://dx.doi.org/10.1007/3-540-34783-6 43

[19] Y. Wang and J. Vassileva, "Bayesian network-based trust model," in WI '03: Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence. Washington, DC, USA: IEEE Computer Society, 2003, p. 372.

[20] Levent V. Orman, "Bayesian Inference in Trust Networks", Cornell University

[21] Shlomo Zilberstein: Using Anytime Algorithms in Intelligent Systems. AI Magazine 17(3): 73-83 (1996)

[22] E. Horvitz, "Problem-Solving Design: Reasoning about Computational Value, Tradeoffs, and Resources", BMIR, 1987

[23] H. Tosun, J. Sheppard, "Incorporating Evidence into Trust Propagation Models Using Markov Random Fields", PERCOM, 2011

[24] D. Koller and N. Friedman, Probabilistic Graphical Models; Principles and Techniques. The MIT Press, 2009

[25] trustlet.org, "Extended epinions dataset. www.trustlet.org," 2009.

[26] GroupLens. Movie Ratings Dataset – MovieLens

[27] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford,Tech. Rep., 1998.

[28] P. Massa and P. Avesani, "Trust metrics on controversial users: Balancing between tyranny of the majority and echo chambers," in International Journal on Semantic Web and Information System 3, no. 1, 2007.

[29] R. Kindermann and J. L. Snell, Markov Random Fields and Their Applications. American Mathematical Society, 1980.