Trust-Based Ant Recommender (T-BAR)

Abdelghani Bellaachia
School of Engineering and Applied Science
The George Washington University
Washington, DC, USA
bell@gwu.edu

Deema Alathel
School of Engineering and Applied Science
The George Washington University
Washington, DC, USA
atheld@gwmail.gwu.edu

***Abstract*—Recommender Systems suggest to users items that may be of interest to them. Collaborative filtering recommender systems suggest the items based on the item ratings provided by similar users in the network. Trust-based recommender systems utilize an explicitly issued trust between users to increase the accuracy of the recommendations. In this paper, we propose a bio-inspired algorithm, called Trust-based Ant Recommender (T-BAR), to further increase the accuracy and the coverage of the recommendations in trust-based networks. T-BAR uses the Ant Colony System computational model to imitate the behavior of ants during their search for a good food source. T-BAR's advantage over other known algorithms is that it considers all the target item ratings along the paths rather than just using the ratings found at the end of each path. The Epinions.com dataset was used for the empirical evaluation of Trust-based Ant Recommender and proved its success by drastically improving the coverage of the recommendations while maintaining a reasonable level of accuracy of the results. T-BAR outperforms the basic CF algorithm that uses the Pearson Similarity and Massa's MoleTrust (MT) by achieving a balanced trade-off between accuracy and coverage.**

Keywords-Ant colony system; Ant colony optimization; Artificial agents; Bio-inspired algorithm; Recommender systems; Trust.

I. INTRODUCTION

In the past few years there has been an increasing demand for personalizing users' experiences on the web and thus for filtering the vast amount of information available online in order to deliver the right piece of information to the right user. Recommender Systems (RS) can assist in providing an adaptive web environment by suggesting to a user items, such as movies, books, music, jokes, articles, etc., that the user may find useful or interesting [13]. Collaborative filtering (CF) techniques in RSs aim to find users that are similar to the active user and then base the recommendation of items on the item ratings provided by those like-minded users [15]. But due to the inherent problems with CF RSs, such as cold start users and the lack of users' ratings, several researchers such as Massa et al. [10] and Golbeck [5] shifted their attention to trust-based recommender systems (TBRS) where users explicitly express how much they trust other users rather than rely on the system to implicitly predict the similarity between them. Massa et al. show in [10] through empirical evaluation how their proposed trust metric MoleTrust can outperform traditional CF RSs. Their trust metric is an algorithm that propagates the explicit trust values over the trust network to predict the trust level among distant users (i.e. the trustworthiness of friends-of-friends). In their application, the explicitly expressed trust values replace the similarity values in traditional CF RSs.

II. RELATED BACKGROUND

A. Recommender Systems

In recent years recommender systems emerged as the most effective solution for creating an adaptive web environment. Due to the increased information overload in the past years there has been high demand, especially by e-commerce websites, to quickly filter the information to suit the needs of different users and therefore to personalize a user's experience on the web. RSs at their core accomplish a single main goal: suggest to users items that are most likely to be of interest to them. This general goal allows RSs to be applied in different domains to recommend movies, music, articles, jokes, and even products (just to name a few). The suggested items should aid a user in making an informed decision about the items, such as whether to see a movie, download a song, read an article or buy a product. Over the years different recommendation techniques have been developed but they usually fall into one of the following classes [14]: content-based filtering (CBF), collaborative filtering (CF), demographic filtering (DMF), knowledge-based filtering, and hybrid recommender systems.

In this paper we compare our work to CF RSs in which the system recommends to the active user items that were liked by likeminded users in the past. Thus the only input needed is the item ratings information, unlike other RS techniques. For instance in DMF personal user information is needed to suggest items based on the active user's demographic profile and in CBF additional information about the items is required to recommend items similar in nature to the ones that the active user liked in the past.

B. Trust-Enhanced Recommender Systems

Trust-enhanced RSs started to gain popularity lately under the belief that people tend to trust the taste of people they know (friends) rather than relying on a RS that tries to find likeminded *unknown* people in the network [16]. In such systems users explicitly express the degree of trust they have with respect to other users in the network, therefore this kind of trust is subjective (unlike the similarity between users): if user *a* trusts user *b* that does not necessarily mean that user *b* trusts user *a*. Another important aspect in such systems is that trust values can be propagated in order to recommend not only items from the active user's web of trust (WOT) but rather from the user's WOT and from others trusted by members of the WOT

(extended WOT) [14]. Trust-enhanced RSs can be based on any of the mentioned RS classes, but for the purpose of this paper, we are interested in Trust-enhanced RSs based on CF techniques. In such systems a required input (in addition to the item ratings) is the trust network; i.e. a list of each user's *trusted* friends (web of trust) and in some cases the trust level of each friend. To further utilize the trust information available, several trust metrics have been proposed in the literature like Golbeck's TidalTrust [5] and Massa et al.'s Trust-aware RS [10]. Trust metrics are basically algorithms that propagate the trust over the trust network in order to determine how much would a user trust other users that are connected to her through a network of friends (i.e. friends-of-friends). By propagating the trust the user is exposed to a wider range of users rather than restricting the recommendations from her direct WOT.

C. Ant Colony System

Ant Colony System (ACS) is one of the ACO algorithms family that falls under swarm intelligence algorithms. ACO was first proposed by Dorigo in 1992 [3]. ACO is a probabilistic technique for solving optimization problems by mimicking the foraging behavior of ants when they are searching within their colony for a good food source. In the context of RSs, we can think of the active user as the ants' nest and of the users with a rating for the target item as food sources. In such scenarios, artificial ants are dispensed from the active user into the network to imitate the foraging behavior of real ants in search for a good food source. Just like real ants, the artificial ants search the solution space and update the pheromone level on traversed paths, increasing the pheromone level on paths leading to *good* sources. The pheromone update process increases the probability of subsequent ants choosing the *good* paths that lead to the good food sources and decreases their probability of choosing the other paths. In this manner, the system moves from an unstable stage, where no path is necessarily better than the other, to a stable one where certain path(s) emerge as being the best ones leading to the *best* food sources.

III. Proposed Trust-Based Ant Recommender

In this section we present our bio-inspired algorithm, Trust-Based Ant Recommender (T-BAR). T-BAR is a dynamic algorithm based on the probabilistic methodology of ACO algorithms, which proved its ability to recommend items to users with good results that balance the trade off between accuracy and coverage.

A. Trust Network and Input Representation

The trust network is modeled as a digraph *G=(V, E)*, where the set of nodes *V* represents users and the set of directed edges *E* represents the trust statements issued between the users. The values on the edges indicate the issued trust values. Figure 1 shows an example of a trust network. In our system the input consists of two matrices: 1) an [*N*x*M*] *item-ratings matrix* that holds the ratings given by the users to different items in the past, with the *N* rows being the users in the system and the *M* columns representing the items in the system; and 2) an [*N*x*N*] *user-trust matrix* that holds the trust statements issued between the *N* users in the system with the rows representing the source users issuing the trust statements and the columns being the target users whom the trust statements are issued for.

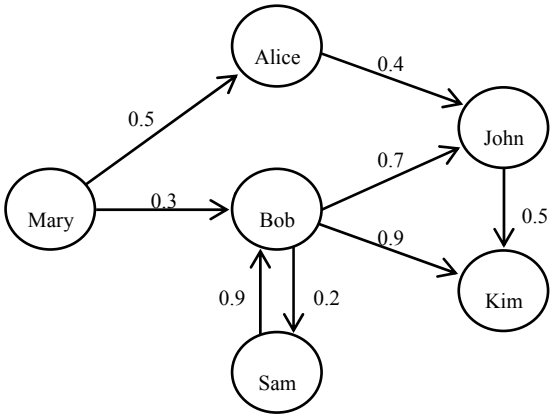

Figure 1. Example of a trust network.

B. T-BAR Specifications

Our bio-inspired algorithm T-BAR is derived from the ACS algorithm proposed by Dorigo et al. in [4].

1) The Artificial Ants and Edge Selection: T-BAR dispatches a predetermined number of *k* ants from the active user with the goal of reaching as many good users as possible. Good users are users within the active user's extended WOT and have ratings *r* for the target item *i*. Each of the k ants moves in the network by first calculating the probability p^k_{xy} of crossing the edge connecting the current node x to each neighboring node y using the following equation:

$$p^k_{xy} = \frac{(\tau_{xy})^{\alpha}(\eta_{xy})^{\beta}}{\sum_{z \in N^k_x}(\tau_{xz})^{\alpha}(\eta_{xz})^{\beta}} \quad (1)$$

where N^k_x refers to the feasible neighborhood of ant *k* when being at node *x*, and where *y* is part of that neighborhood.

Note that the probability p^k_{xy} highly depends on two parameters in (1): τ_{xy} which is the pheromone level on the edge *xy*, and η_{xy} which is the desirability of the move from node *x* to node *y*. α and β are parameters that control the influence of τ_{xy} and η_{xy}. It is worthwhile to note that in ACO algorithms in general, η_{xy} is considered as a priori *desirability* computed by heuristics while τ_{xy} is a posteriori indication about the *goodness* of the move. Once the probabilities p^k_{xy} are calculated the ant *k* moves along the edge that yielded the highest probability.

In T-BAR the ants stop their solution construction either when each reaches a set search depth *d* or if no more edges can be traversed. Once all *k* ants stop, the process repeats by dispatching the *k* ants again from the active user. The repeated process stops completely after a certain number of iterations *t*. During the last iteration each ant *k* keeps a record of *good* users that it came across while constructing its last path. Remember that *good* users are those with a rating *r* for the target item *i*.

We define the parameters of (1) in T-BAR as follows:

$$\eta_{xy} = T_{xy} \tag{2}$$

$$\beta = P_y \tag{3}$$

$$\alpha = 1 \tag{4}$$

where T_{xy} is the trust value issued from user x to user y, and P_y is the popularity of user y computed as the average trust issued to user y by users z that have user y in their WOT_z:

$$P_y = \frac{\sum_{y \in WOT_z} T_{zy}}{n(z)} \tag{5}$$

where $n(z)$ is the number of users z. Typically in ACS α is usually set to 1 [4].

2) Pheromone Update Mechanism: In real life, while ants forage for food they deposit pheromone along their path to inform other ants that the path has been discovered. Other ants can smell the pheromone on the paths and subsequently tend to follow, probabilistically, the paths with a higher pheromone concentration. The pheromone build-up on the path leading to a good food source results in all the ants at the end converging to that path. But an important trait about pheromones is that they evaporate as time passes by. The evaporation mechanism reduces the influence of the pheromone deposited by early ants and favors the exploration of new paths. In other words, the evaporation prevents the ants from converging to poor paths discovered during the early stages of the search.

In ACS the pheromone model is accomplished on two levels: on a local level and on a global level.

In T-BAR, the *local pheromone update* occurs as each ant k traverses an edge xy. The pheromone level τ_{xy} is adjusted by:

$$\tau_{xy} = (1-\rho).\tau_{xy} + \rho.\tau_{xy}^0 \tag{6}$$

where τ_{xy} is the pheromone level on the edge xy, ρ is the pheromone evaporation coefficient, and τ^0_{xy} is the initial pheromone level on the edge xy. Typically in ACS ρ is usually set to 0.1 [4].

The *global pheromone update* in ACS takes place at the end of each iteration when all the k ants finish constructing their solutions. Unlike the local pheromone update, not all edges will be globally updated but rather only the ones belonging to the best solutions (paths) constructed in that iteration. In T-BAR the global pheromone update is accomplished in several steps. First we compute the path trust PT_k for each constructed solution by an ant k. The *path trust* [12] is a function of the number of co-rated items $n(I_{xy})$ between two adjacent users x and y and the trust T_{xy} issued by user x towards user y:

$$PT_k = \frac{\sum_{xy \in P_k} \left(n(I_{xy}).T_{xy} \right)}{\sum_{xy \in P_k} n(I_{xy})} \tag{7}$$

where P_k refers to the path constructed by ant k. After calculating the k path trusts, we add the paths P_k that satisfy the criteria $PT_k \geq PT_{threshold}$ to the set of best paths P^{best}. Lastly, we apply the global pheromone update on the edges belonging to the paths in P^{best}:

$$\tau_{xy} = (1-\rho).\tau_{xy} + \rho.\Delta\tau_k^{best} \tag{8}$$

where $xy \in P_k$ and $P_k \in P^{best}$ and:

$$\Delta\tau_k^{best} = PT_k \tag{9}$$

Note that the global pheromone update in ACS contributes to the pheromone build-up on *good* paths and thus helps the ants in subsequent iterations to ultimately converge to these paths. Also note that by using the number of co-rated items between users we are incorporating a CF technique in the process. Although what we used is not a similarity measure, but we like to think of the number of co-rated items as a semi-similarity measure.

3) Initial Pheromone Level on Edges: In ACO algorithms the initial pheromone level is usually set to a value close to the amount of pheromone expected to be deposited in any iteration. A careful choice of this value guarantees that the system will converge in a reasonable amount of time to the best paths. Choosing a small value will cause the convergence to be too slow, so by the time the last iteration is done, no obvious path will emerge as being the best in which case we would need additional iterations to find a better path. On the other hand picking a larger value for the initial pheromone level will cause the system to converge early to a path constructed in the early iterations which is not necessarily the best solution but rather a sub-optimal one.

The initial pheromone level in ACO is usually initialized using a carefully chosen constant or using a pre-calculated value obtained from running a quick sub-optimal path construction algorithm [4], such as the nearest-neighbor algorithm, on the network. In such scenarios the pheromone level on all edges in the system are initialized using the same value τ_0.

In T-BAR though we propose to initialize an edge locally right before an ant encounters it (regardless whether it is traversed or not) using the trust information in the network. Before an ant k computes the probability p^k_{xy} of traversing adjacent edges xy, it checks whether they have been initialized or not. If not then the adjacent edges are initialized using the inverse of the sum of their assigned trust values T_{xy} from node x:

$$\tau_{xy}^{0} = \frac{1}{\sum_{z \in WOT_x} T_{xz}} \quad (10)$$

Therefore, the edges in the system will not necessarily have the same initial pheromone level. We believe that our proposed approach of initializing the pheromone level locally has several advantages: instead of using the information in the whole network to initialize an edge we use local information which reduces the number of external factors (from distant nodes) that could affect the initial pheromone level; also in such networks, not all edges are necessarily traversed by the ants so with our proposed initialization we reduce the number of unnecessary initializations. In addition we believe that by the end of the last iteration the uninitialized edges can be further analyzed to extract useful information that may assist in improving the performance of the system in the long run.

C. Predicting the Rating for the Target Item

In our previous discussion of artificial ants and edge selection, we mentioned that the k ants keep a record of the *good* users that they come across while constructing the paths in the last iteration. The *good* users are users u that have a rating for the target item i, denoted by r_i^u.

The reason behind keeping the record is that in T-BAR at the end of the last iteration the system should have converged into the best solutions (paths). T-BAR then averages the ratings r_i^u obtained from users y found on these paths to calculate the target item's predicted rating for the active user x:

$$r_i^x = \frac{\sum_{x \in P^{best}} r_i^u}{n(u)} \quad (11)$$

where $n(u)$ is the number of users u. We believe that since we labeled the paths as being good ones and since they have a high path trust value we might as well use all the ratings encountered along them as opposed to only using the item rating given by the last user reached on each path, which is the case in both TidalTrust [5] and MoleTrust [10].

IV. Experimental Evaluation

In this section we provide the details of the experiments that were conducted to validate the performance of our proposed algorithm, *Trust-Based Ant Recommender*. We compare our results to the ones obtained by Massa et al. in [10] since it is one of the major techniques that were applied to trust-enhanced RSs [1, 6, 11, 17]. Namely we compare our results to a basic CF algorithm that uses the Pearson Similarity measure as the similarity between the users and to Massa's proposed *MoleTrust* algorithm which replaces the similarities in CF with the explicit trust values in the network.

A. The Epinions Dataset

The Epinions dataset was used for our empirical evaluation. The reason behind our choice lies in the fact that there is a scarcity in the availability of datasets that contain both item ratings along with explicitly issued trust values among the users. The dataset is composed of 49,290 users that rated 139,738 unique items at least once. The ratings range between 1 and 5 with 5 being the best rating. In addition there are 487,181 explicitly issued trust statements among the users. In this dataset there is no range for the issued trust values; if a user trusts another then that is expressed with a trust value of 1. When it comes to the ratings 45% of them are 5 and 29% of them are 4, which means that more than half the ratings are good ones.

The dataset can be further classified into two major categories, or *views*. These views are [10]: 1) *cold start users*, users who rated less than 5 items; and 2) *heavy raters*, users who rated more than 10 items. More than half the users in the dataset rated less than 5 items each and thus the dataset has a substantial number of cold start users. Such users typically make it harder for RSs to predict new item ratings for them due to lack of information.

B. T-BAR's Parameters

We applied the *leave-one-out* technique to test T-BAR's ability to correctly predict the items' ratings. Just like in any ACO algorithm, we had to experiment with the different parameters in T-BAR in order to determine the best set to be used. We conducted a total of 22 experiments varying the number of ants k (5, 10, 20, 30, 40, 50), the number of iterations t (5, 10, 20, 30, 40, 50), the search depth d (10, 20, 30, 40, 50), and the path trust threshold $PT_{threshold}$ (0.1, 0.3, 0.5, 0.7, 0.9). The baseline when we tested all these variations was 10 ants, 10 iterations, a search depth of 30, and a path trust threshold of 0.5. It turned out that there was not a significant improvement in the *Mean Absolute Error* (MAE) or the *ratings coverage* (RC) when we varied the parameters, thus we chose to use the mentioned baseline and to just vary the search depth d (10, 30, 50) for comparing T-BAR with the results presented in [10].

C. Evaluation Metrics

In order to be capable of comparing our results to the ones obtained in [10], we used the *Mean Absolute Error* (MAE) and the *ratings coverage* (RC) as the evaluation metrics.

The MAE is the average of the absolute difference between the expected rating and the actual rating for the target item.

The *ratings coverage* (RC) [7] assess a RS's ability to generate a prediction for the hidden rating, regardless of its accuracy. RC refers to the fraction of ratings that were generated by the RS, while using the leave-one-out technique, against the actual number of ratings in the dataset. In other words, it is a measure of a RS's ability to predict a rating for a target item.

D. Experimental Results

We compare our results to 2 different algorithms presented in [10]: *CF* which is a basic CF algorithm that uses the Pearson Similarity and *MT* which is Massa's proposed MoleTrust algorithm but with three different propagation horizons (1, 2, and 3) referred to as MT1, MT2, and MT3. Tables 1 and 2 show the results obtained by T-BAR against the two algorithms. T-BAR10, T-BAR30, and T-BAR50 refer to the three

TABLE I. MAE OF THE ALGORITHMS ON DIFFERENT VIEWS

Views	Algorithm						
	CF	MT1	MT2	MT3	T-BAR 10	T-BAR 30	T-BAR 50
All	0.843	0.832	0.846	0.829	0.298	0.315	0.304
Cold Start Users	1.094	0.674	0.833	0.854	1.459	1.4	1.426
Heavy Raters	0.850	0.873	0.869	0.846	0.212	0.22	0.22

TABLE II. RC OF THE ALGORITHMS ON DIFFERENT VIEWS

Views	Algorithm						
	CF	MT1	MT2	MT3	T-BAR 10	T-BAR 30	T-BAR 50
All	51.3%	28.3%	60.5%	74.4%	93%	97%	97%
Cold Start Users	3.2%	11.1%	25.1%	41.7%	91%	95%	96%
Heavy Raters	57.5%	30.9%	64.8%	77.8%	93%	97%	97%

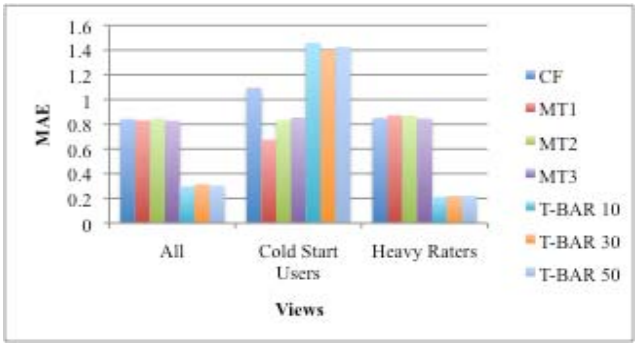

Figure 2. MAE across different views.

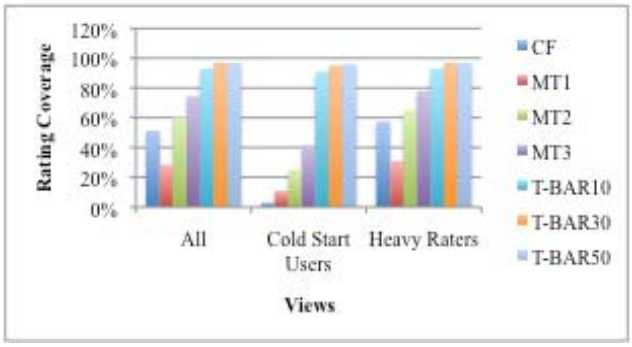

Figure 3. RC across different views.

different search depths used. To make more sense of the comparison it is worth noting that MT follows a breadth-wise search while T-BAR searches for a solution in a depth-wise manner. It would not be fair to test T-BAR using the same depths used by MT due to the different ways that the two algorithms behave. Doing so would cause T-BAR to traverse a very few number of nodes/edges compared to the ones reached by MT.

A quick glance at the first row in Tables 1 and 2 will show that T-BAR drastically increases the overall accuracy and coverage of the recommendations. Recall that more than half the users in the Epinions dataset are classified as cold start users which usually poses a challenge for RSs, yet T-BAR manages to achieve a low MAE in general. Figures 2 and 3 illustrate T-BAR's overall improvement over MT and CF with respect to both accuracy and coverage. When considering the performance of the different algorithms with only cold start users T-BAR does not perform that well in providing an accurate rating prediction but again the RC is way better than those obtained from MT. On the other hand though when considering heavy raters we can see that T-BAR has a significant improvement over the other algorithms with respect to both the prediction accuracy and the coverage. For heavy raters T-BAR had a MAE of ~0.2 and at least 93% RC compared to MT3's MAE of 0.846 and RC of 77.81%. The reason behind T-BAR's superior performance for heavy raters can be attributed to the way T-BAR works. Recall from our earlier discussion that at the end of each iteration the number of co-rated items between adjacent users along P_k plays a major role in increasing the path trust and thus increasing the pheromone level on that path which will also increase the probability of having a greater number of co-rated items between adjacent users. In addition these paths were selected by the ants while constructing their solution partly because of their high trust along their edges. We believe that these two reasons resulted in the increase in the predictions' accuracy.

We believe that T-BAR's ability to incorporate all the trusted users' ratings for such items along the paths over several iterations resulted in its superior performance.

E. Summary of Results

The empirical evaluation of our proposed algorithm T-BAR on the Epinions dataset proved that the algorithm can achieve a good balance between prediction accuracy and ratings coverage. The novelty of T-BAR proved to be especially useful for heavy raters where we managed to achieve a MAE as low as ~0.2 for such users as opposed to MT and CF algorithms' MAE of ~0.8. On the other hand, MT and classic CF techniques outperformed T-BAR with respect to cold start users but with the downside of drastically reducing the coverage.

We believe that T-BAR can be useful in such cases for cold start users because there are situations where the system can tolerate bad predictions early on for those users until they become heavy raters. Recall that we defined heavy raters as users with more than 10 ratings, which is not a big burden for the average user to achieve in most recommender systems. For example consider Netflix.com where users are encouraged to rate as many movies as they can so that the system can provide better rating predictions for unseen movies. If CF or MT techniques were to be applied in such scenarios the system would be able to provide a rating for unseen movies only for ~3% of cold start users in the case of CF algorithms, and for ~42% of such users in MT algorithm's best RC (MT3). On the other hand T-BAR would be able to provide a rating for at

least 91% of cold start users in hopes of encouraging them to rate more movies until they become heavy raters and start receiving more accurate rating predictions. In critical systems though where the accuracy of the prediction outweighs the importance of the coverage, CF techniques would definitely be the better option to be applied rather than MT or T-BAR.

V. CONCLUSION

In this paper we proposed the application of the ant colony system algorithm to trust-enhanced recommender systems in addition to incorporating collaborative filtering techniques in the process. We named our approach T-BAR. The goal was to increase the accuracy of the ratings' prediction and system coverage. The novelty of T-BAR lies in the fact that it is probably the first successful application of an algorithm from the swarm intelligence field to the area of trust-enhanced recommender systems. One of the major advantages of T-BAR is that unlike other trust metrics it considers all the item ratings that it encounters along the path rather than just using the final item rating that was reached. We have discussed the application of our proposed algorithm T-BAR to a real-world large dataset with the empirical evaluation and comparison of the results obtained against some known algorithms. We also analyzed the results with respect to different views of the dataset in which the users and the items were further classified to better understand T-BAR's performance in different situations. In general the empirical results show that T-BAR always provide a significantly better coverage of the dataset when compared to the other algorithms, regardless of the prediction accuracy. This property can be useful in systems in which we are interested in achieving a higher quantity of predictions over the quality of these predictions.

Furthermore T-BAR achieves a significantly low MAE when considered over the whole dataset and over heavy raters, which indicates that T-BAR would outperform MT and traditional CF techniques when applied to datasets where cold start users are a minority in the system such as the MovieLens dataset.

Based on our obtained results and the comparisons we performed we can conclude that T-BAR is the better choice for recommending items in datasets composed mainly of heavy raters.

At this point we still wish to tackle cold start users in hopes of achieving better performance results. We will try to alter different features in T-BAR, such as altering the values of the influence parameters α and β, and we will monitor the effects of the changes on the results. In addition since T-BAR seems to perform well for heavy raters, we plan to apply it to a dataset that is composed mainly of heavy raters such as MovieLens. Since such datasets do not incorporate trust values into the system, the challenge will be in how to properly alter T-BAR so that we can still achieve the same level of results obtained when we applied it on the Epinions dataset.

REFERENCES

[1] P. Avesani, and P. Massa, "Moleskiing.it: A Trust-aware recommender system for ski mountaineering," International Journal for Infonomics, pp. 1-10, 2005.

[2] R. Burke, "Hybrid web recommender systems," in B. Peter, K. Alfred, and N. Wolfgang (Eds.): The Adaptive Web, Springer-Verlag, pp. 377-408, 2007.

[3] M. Dorigo, "Learning and natural algorithms," Doctoral Dissertation, Politecnico di Milano, 1992.

[4] M. Dorigo, and T. Stützle, "Ant colony optimization", MIT Press, 2004.

[5] J. Golbeck, "Computing and applying trust in web-based social networks," Doctoral Dissertation, University of Maryland at College Park, 2005.

[6] J. Golbeck, "Generating predictive movie recommendations from trust in social networks", In K. Stølen, W. H. Winsborough, F. Martinelli, and F. Massacci (Eds.): Lecture Notes in Computer Science, vol. 3986, pp. 93-104, 2006.

[7] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Transactions on Information Systems, vol. 22, pp. 5-53, 2004.

[8] T. Mahmood, and F. Ricci, "Improving recommender systems with adaptive conversational strategies," Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, Torino, Italy, 2009, pp. 73-82.

[9] P. Massa, and P. Avesani, "Trust-aware collaborative filtering for recommender systems," Proceedings of the Federated International Conference on the Move to Meaningful Internet, Larnaca, Cyprus, October 2004, Springer-Verlag, pp. 492-508.

[10] P. Massa, and P. Avesani, "Trust-aware recommender systems," Proceedings of the 2007 ACM Conference on Recommender Systems, Minneapolis, MN, USA, 2007, pp. 17-24.

[11] J. O'Donovan, and B. Smyth, "Trust in recommender systems," Proceedings of the 10th International Conference on Intelligent User Interfaces, San Diego, California, USA, 2005, pp. 167-174.

[12] M. Papagelis, D. Plexousakis, and T. Kutsuras, "Alleviating the sparsity problem of collaborative filtering using trust inferences," Proceedings of the Third International Conference on Trust Management, Paris, France, 2005, pp. 224-239.

[13] P. Resnick, and H.R. Varian, "Recommender systems," Commun. ACM, vol. 40, pp. 56-58, 1997.

[14] F. Ricci, L. Rokach, A.B. Shapira, and A.P.B. Kantor, "Recommender systems handbook," Springer-Verlag, New York, NY, USA, ch. 2, 4, 2010.

[15] J.B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in B. Peter, K. Alfred, and N. Wolfgang (Eds.): The Adaptive Web, Springer-Verlag, pp. 291-324, 2007.

[16] R.R. Sinha, and K. Swearingen, "Comparing recommendations made by online systems and friends," Proceedings of the DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries, 2001.

[17] P. Victor, C. Cornelis, M.D. Cock, and A.M. Teredesai, "Key figure impact in trust-enhanced recommender systems', AI Commun., vol. 21, pp. 127-143, 2008.