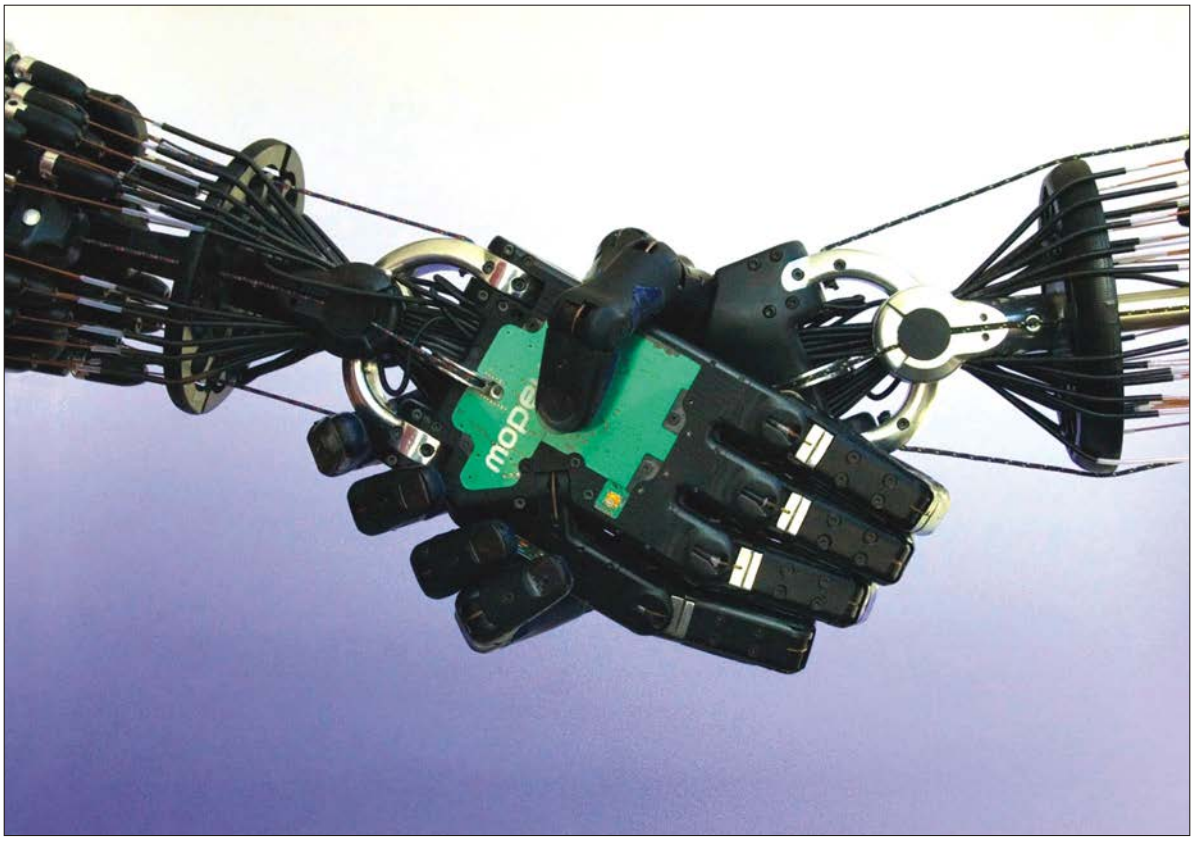

Journal of Artificial Intelligence

ISSN 1994-5450

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Artificial Intelligence 2 (2): 40-55, 2009
ISSN 1994-5450

Recommender System Based on Collaborative Behavior of Ants

[1]P. Bedi, [1]R. Sharma and [2]H. Kaur
[1]Department of Computer Science, New Academic Block,
Adjoining Arts Faculty Building, University of Delhi, Delhi-110007, India
[2]Department of Computer Science, Hans Raj College,
University of Delhi, Delhi-110007, India

Abstract: This study uses collaborative filtering approach and proposes an Ant Recommender System (ARS) based on collaborative behavior of ants for generating Top-N recommendations. Present proposed system ARS works in two phases. In the first phase, opinions from users collected in the form of user-item rating matrix are clustered offline using ant based clustering algorithm into predetermined number of clusters and stored in the database for future recommendations. In the second phase, the recommendations are generated online for the active user. The pheromone updating strategy of ants is combined with similarity measure for choosing the clusters with good quality ratings. This helps in improving the quality of recommendations for the active user. The performance of ARS is evaluated using Jester dataset available on the website of University of California, Berkeley and compared with traditional collaborative filtering based recommender system.

Key words: Swarm intelligence, ant colony optimization, recommender system, collaborative filtering, pheromone updating

INTRODUCTION

Recommender Systems have gained wide-spread acceptance and attracted increased public interest during the last decade, leveling the ground for new sales opportunities in E-Commerce (Herlocker *et al*., 1999; Adomavicius and Tuzhilin, 2005). They serve as crucial tools to overcome the information overload, sifting through very large sets of information and selecting information that is relevant for the active user. The term "active user" refers to the person for whom recommendations are made.

Recommender systems work by collecting data from users, using explicit and/or implicit methods and then comparing the collected data to the active user data to find a list of recommendations for the active user. The prevalent three approaches to building recommender systems are collaborative filtering, content based approach and hybrid methods. Collaborative Filtering (CF) collects opinions from users in the form of ratings on various items. The recommendations produced are based only on the opinions of users similar to the active user (neighbors). Content based approach suggests items that are similar to the ones the active user has shown a preference for in the past rather than on the preferences of other users. Content based matching requires a representation of the items in terms of features (attributes). To improve performance, mostly CF is combined with one or more recommendation techniques i.e., content based, knowledge-based, demographic or utility based is called hybrid recommender systems.

Corresponding Author: Dr. Punam Bedi, Department of Computer Science, New Academic Block,
Adjoining Arts Faculty Building, University of Delhi, Delhi-110007, India
Fax: 011-27662553

Collaborative Filtering (CF) takes its roots from something humans have been doing for centuries viz., sharing opinions with others. It brings together the opinions of large interconnected communities on the web (Schafer *et al*., 2007). Many collaborative systems have been developed in the academia and the industry. Using stereotypes, the Grundy system was developed to build individual user models and use them to recommend relevant books to each user (Rich, 1979). Later on, the Tapestry system relied on each user to identify like-minded users manually (Goldberg *et al*., 1992). GroupLens (Konstan *et al*., 1997; Resnick *et al*., 1994) in the domain of Usenet newsgroup articles, Bellcore's video recommender (Hill *et al*., 1995) in the domain of movies and Ringo (Shardanand and Maes, 1995) in the domain of music and musical artists were the first systems to use collaborative filtering algorithms to automate predictions. Other examples of collaborative recommender systems include the book recommendation system from Amazon.com, the PHOAKS system that helps people find relevant information on the WWW (Terveen *et al*., 1997) and the Jester system that recommends jokes (Goldberg *et al*., 2001). Automated collaborative filtering approach exposing the reasoning and data behind the recommendation is explained by Herlocker *et al*. (2000). Different item-based recommendation algorithms are analyzed by Sarwar *et al*. (2001). Walker *et al*. (2004) reviews collaborative filtering approach to recommendations and also implemented Altered Vista. Huang and Zeng (2005) represent input data as a bipartite graph and develops its topological measures to capture patterns that exist in the input data relevant to recommendations. Schafer *et al*. (2007) discusses the core concepts of collaborative filtering, its primary uses for users of the adaptive web, the theory and practice of CF algorithms, design decisions regarding rating systems and acquisition of ratings. Researchers have also applied Swarm Intelligence techniques to recommender systems. Ujjin and Bentley (2003) have described Particle Swarm Optimization (PSO) recommender system in which PSO algorithm has been employed to learn personal preferences of users and provide tailored suggestions. They demonstrated that PSO system outperformed a non-adaptive approach based on the Pearson algorithm and obtained higher prediction accuracy than the Genetic Algorithm system and Pearson algorithm in most cases. A system called CASIS (Lorenzi *et al*., 2005) has been developed combining case based reasoning approach with a metaphor from colonies of social insects, namely the honey bee dance. This combination has been used in the retrieval step of the recommendation cycle. Web-based system user interface hybrid recommendation method based on ant colony metaphor is presented in (Sobecki, 2008). The applied hybrid recommendation method employs demographic, content based and collaborative filtering approaches. In this study, ant colony metaphor is used for selecting the most optimal path in the user interface graph that specifies the user interface parameters for the specified user.

Clustering algorithms have been used by researchers to quickly locate a user's neighbors. Clusters of users similar to the active user are quickly discovered and nearest neighbors can be selected from the most similar clusters. A variety of clustering techniques exist in the literature, but k-means is commonly used clustering technique. The k-means algorithm implementation for cluster analysis as data mining approach has been discussed in several research papers (Kuo *et al*., 2005; Vrahatis *et al*., 2002). Saatchi and Hung (2005) described hybridization of the ACO with k-means algorithm to solve image clustering problems. Shelokar *et al*. (2004) presents an ACO methodology for optimally clustering objects and compares the performance of the algorithm with other popular heuristic methods viz. genetic algorithm, simulated annealing etc. They claim that, ant based clustering algorithm optimally clusters N objects into K clusters and their computational simulations reveal very encouraging results in terms of the quality of solution found, the average number of function evaluations and the processing time required. The data clustering technique developed by Shelokar *et al*. (2004) is applied by Kekec *et al*. (2006). Handl and Meyer (2007) have categorized ant based clustering methods into two main groups viz., (1) methods that directly mimics the clustering behavior observed in real ant colonies and (2) methods that are less directly inspired by nature i.e., the clustering task is reformulated as an optimization task and general purpose ant based optimization heuristics are utilized to find good or near-optimal clustering.

The success of the collaborative filtering technology depends on the process of locating people with similar profiles (neighbors). We believe that an approach for generating recommendations based on the biological metaphor of ant colonies (Dorigo *et al*., 1996; Dorigo and Stutzle, 2004; Blum, 2005) is a promising research area. In the real ant systems, ants tend to lay a substance called pheromone while walking from their nests to food source and vice versa. Ants are attracted by pheromones coming from fellow type ants and repulsed by pheromone of non-fellow type ants. Paths that are marked by stronger amount of pheromone are chosen with higher probability than those that have weaker amount of pheromone deposit. This collaborative behavior between identical type ants has an analogy with the collaborative world as people mostly collect opinions from their like-minded friends, neighbors etc.

One of the main limitations of CF based recommender systems is that it provides recommendations based solely on the opinion of the users whose preferences best matches the taste of active user. A case may arise when an item has not been rated well in the cluster whose similarity matches best with active user but is rated well in other clusters which have similarity closer to his taste. In such a scenario, combining pheromone information with similarity measure for choosing clusters provides active user with good set of alternative recommendations. In addition to his taste, clusters that are marked by stronger amount of pheromone have the higher probability of being chosen than those that have weaker amount of pheromone deposit. Also, one of the fundamental challenges for recommender systems is to improve the quality of the recommendations. In such a scenario, pheromone updating step guides the search to a better recommendation. The positive feedback in the form of pheromone deposition results in achieving an emergent, unified behavior for the recommender system as a whole and produces a robust system capable of finding improved quality recommendations. In this study, we have tried to improve the quality of recommendation based on the pheromone density and pheromone updating strategy.

PROPOSED ANT RECOMMENDER SYSTEM

Ant Recommender System (ARS) works in two phases. Phase I is the preprocessing phase, which is offline. In this phase, the background data in the form of user-item rating matrix is collected and clustered using an ant based clustering algorithm into predetermined number of clusters. Once the clusters are obtained, the cluster data along with their centroids are stored in the database for future recommendations. Phase II is online in which the recommendation process takes place for the active user. Here, the pheromone deposition/evaporation technique known from ant algorithms is combined with similarity measure for choosing the best clusters for making the recommendations. Rating quality of each item unrated by active user is computed in the chosen clusters. To generate recommendations, clusters are further selected from the chosen clusters based on rating quality of item.

Recommendations are then made by computing the weighted average of the ratings of items in the selected clusters. Pheromone is updated based on recommendations made to active user and past recommendations. This helps in improving the quality of recommendations for future users. Figure 1 shows the steps where ant colony metaphor has been applied to the traditional collaborative filtering based recommendation process.

The working of ARS is described below in detail with the help of Jester data set example.

Phase I: Preprocessing Phase

Step 1: Normalization of Background Data i.e., Rating Matrix

User-item ratings taken from Jester dataset rated in the scale of -10 to +10 is normalized in the scale 0 to 1, where 0 indicates that the item is not rated by the corresponding user. To ease the discussion, running example shown in Table 1 is used, where U_1-U_{10} are users and J_1-J_{10} are items (jokes) rated/ unrated by the user.

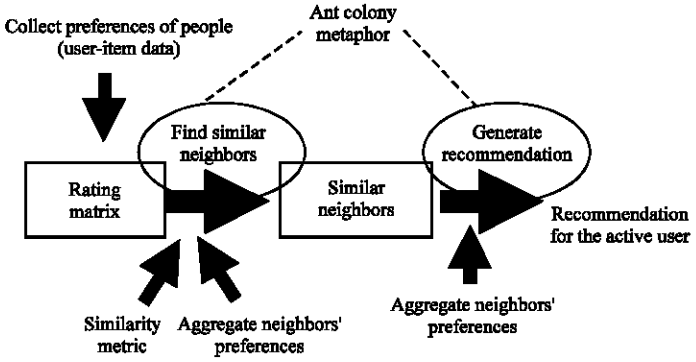

Fig. 1: Ant based collaborative filtering process

Table 1: Running example of rating matrix from Jester dataset after normalization in the range 0 to 1

Users	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
U_1	0.15	0.94	0.06	0.13	0.16	0.11	0.05	0.72	0.09	0.29
U_2	0.71	0.51	0.82	0.73	0.41	0.06	0.48	0.26	0.94	0.96
U_3	0.00	0.00	0.00	0.00	0.95	0.96	0.95	0.96	0.00	0.00
U_4	0.00	0.92	0.00	0.00	0.60	0.91	0.38	0.81	0.00	0.61
U_5	0.92	0.74	0.32	0.26	0.58	0.60	0.85	0.74	0.50	0.79
U_6	0.23	0.35	0.54	0.11	0.18	0.31	0.11	0.48	0.20	0.43
U_7	0.00	0.00	0.00	0.00	0.93	0.05	0.89	0.94	0.00	0.00
U_8	0.84	0.67	0.96	0.22	0.13	0.44	0.96	0.59	0.27	0.31
U_9	0.34	0.35	0.07	0.19	0.10	0.51	0.27	0.09	0.14	0.44
U_{10}	0.66	0.76	0.76	0.66	0.82	0.76	0.94	0.64	0.66	0.91

0 indicates not rated

Step 2: Clustering the Rating Matrix Using ant Based Clustering Algorithm

The rating matrix containing the user-item data is clustered into K clusters using ant based clustering technique (Shelokar *et al*., 2004). Each agent (ant) discovers a possible partition of items in a given dataset using some metric like Euclidean distance. Clustering information of items (jokes) associated with an ant is accumulated in the global information hub (pheromone trail matrix) and is used by other agents to construct possible clustering solution string S and iteratively improve them. Each element of S contains the cluster number which corresponds to one of the items. For example, a solution string S for N (items) = 10 and K (clusters) = 3 is as follows:

2	1	3	1	2	2	1	1	3	1

The first element in S indicates that first item (i.e., N = 1) is assigned cluster number 2, second item is assigned cluster number 1 and so on.

The algorithm executes three steps at any iteration level viz., (1) generation of new R solutions by agents using modified pheromone trail information available from previous iteration, (2) performing local search operation on the newly generated solutions and (3) updating pheromone trail matrix. The algorithm works for a fixed number of iterations and the solution having the lowest fitness function value (F) represents an optimal or near-optimal partitioning of items into subsets in a given dataset.

Clustering the rating matrix helps locate similar user profiles. Some intermediate results during clustering process for dataset shown in Table 1. Table 2 depicts pheromone trail matrix generated during run of ant based clustering algorithm. Table 3 depicts solution string S generated by R = 10 ants along with fitness function F in sorted order of F during run of ant based algorithm. Table 4

Table 2: Pheromone trail matrix generated during run of ant based clustering algorithm

Users	K = 1	K = 2	K = 3
U_1	0.3997	0.0099	0.0099
U_2	0.2735	0.1361	0.0099
U_3	0.3997	0.0099	0.0099
U_4	0.1361	0.0099	0.2735
U_5	0.3997	0.0099	0.0099
U_6	0.2735	0.0099	0.1361
U_7	0.1361	0.2735	0.0099
U_8	0.3997	0.0099	0.0099
U_9	0.1361	0.0099	0.2735
U_{10}	0.3997	0.0099	0.0099

Table 3: Solution String S generated by R = 10 ants along with Fitness function F in sorted order of F during run of ant based clustering algorithm

String	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	F
S_1	1	1	1	3	1	1	2	1	3	1	7.5880
S_2	1	2	1	1	1	3	1	1	1	1	7.9214
S_3	1	1	1	3	1	1	3	1	2	1	8.0135
S_4	1	1	2	2	1	1	1	1	1	1	8.1961
S_5	1	3	1	1	1	1	1	1	1	1	8.5137
S_6	3	1	1	1	1	3	1	1	1	1	8.9393
S_7	1	1	1	2	2	1	2	1	1	1	9.3505
S_8	1	1	1	1	2	1	2	1	1	1	9.4913
S_9	1	1	1	1	2	1	1	1	1	1	9.5892
S_{10}	1	1	1	1	3	3	1	1	1	1	10.023

Table 4: Best solution S_1 obtained after a fixed number of iterations

String	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}
S_1	3	2	1	1	1	3	1	2	3	1

Table 5: Users in each cluster and the centroid

K	Users	Centroid
1	U_3, U_4, U_5, U_7,U_{10}	U_3
2	U_2, U_8	U_2
3	U_1, U_6, U_9	U_6

shows the best solution S_1 obtained after a fixed number of iterations which indicates that U_3, U_4, U_5, U_7 and U_{10} belong to cluster 1, U_2 and U_8 belong to cluster 2, U_1, U_6 and U_9 belong to cluster 3.

Step 3: Computing Centroid of Each Cluster

Once the clusters are created, centroid of each cluster can be computed using Euclidean distance measure. Here, preferences of each user are compared with every other user in the cluster and the user with maximum similarity with other users becomes the centroid of that cluster. Table 5 shows the users in each cluster along with the centroid.

Storing the clusters and their centroid in database for future recommendations are made in step 4.

Step 4: Initializing Pheromone for Each Cluster

Initially, at time t = 1, the pheromone attached to each cluster depends on the density of the cluster i.e., relative number of users in the cluster. If the relative number of users is more, pheromone attached is more and vice versa. The equation for initializing pheromone for cluster i is given below:

$$\text{pher}_i(t) = \frac{\text{No. of users in cluster i}}{\text{total No. of users}} \quad (1)$$

Phase II: Recommendation Process for the Active User

Step 1: Choosing the Appropriate Clusters

The cluster to be chosen depends upon the utility of the cluster which in turn depends on two factors viz., density of cluster and similarity with active user profile. The probability that the cluster i is chosen for generating recommendations at time t is given by Eq. 2.

$$P_i(t) = \frac{pher_i(t).sim_i}{\sum_{i=1}^{K} pher_i(t).sim_i} \quad (2)$$

Where:

Sim_i = Value of the similarity function to measure similarity between the active user profile and the centroid of ith cluster

$pher_i(t)$ = Amount of pheromone associated with ith cluster at time t

K = Total number of clusters

The density of cluster is determined by pheromone value. During phase I, pheromone value is initialized using Eq. 1. Then, when the recommendations are generated, pheromone value is updated based on weighted rating quality of items in the clusters. The similarity function measures the similarity between the active user profile and the centroid of the cluster. There are a number of possible measures for computing the similarity, for example the Euclidean distance metric, cosine/vector similarity and the Pearson Correlation metric. The Euclidean distance between the active user profile and the centroid of the cluster can be given as Eq. 3:

$$Dist(Cent_i, U) = \{\sum_{j=1}^{d} | Cent_{i,j} - U_j |^2\}^{1/2} \quad (3)$$

Where:

d = Dimensionality of data i.e the number of attributes

$Cent_i$ = Centroid of the cluster i

U = Active user profile

$Cent_{i,j}$ = jth attribute of the centroid profile in cluster i

U_j = jth attribute of the active user profile

Therefore, similarity measure is estimated in Eq. 4 as follows:

$$Sim_i(Cent_i, U) = \frac{1}{dist(Cent_i, U)} \quad (4)$$

The similarity measure of the active user profile is calculated with each cluster in order to find a cluster which has users with similar preferences.

The amount of pheromone $pher_i(t)$ available on each cluster incorporates an indirect form of communication suggesting the best cluster to be chosen by the ARS for recommendation. The pheromone updating strategy is explained in detail in subsection further. The clusters whose probability value lies in the range interval ((highest probability-0.1)≤probability≤highest probability) are chosen for generating recommendations for the active user instead of only the cluster with highest probability. This overcomes the limitation of CF based recommender systems where recommendations are provided based solely on the opinion of the users with most similar preferences and provides active

Table 6: Normalized data of active user in the range 0 to 1

	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
Active User	0.38	0.71	0	0	0.25	0	0.64	0.27	0	0.59

Table 7: Process of choosing clusters

	K = 1	K = 2	K = 3
Pheromone value (pher)	0.5	0.2	0.3
Similarity measure	0.0890	0.5475	0.3635
Probability function (P)	0.1692	0.4163	0.4146

Clusters chosen K = 2, 3

Table 8: Rating quality computed for jokes

Jokes unrated by active user	Q (K = 2)	Q (K = 3)
3	0.6114	0.0752
4	0.1310	0.4636
6	0.1489	0.1105
9	0.1085	0.3504

user with good set of alternative recommendations. Ratings given by active user for jokes 1 to 10 (J_1-J_{10}) normalized in the range 0 to 1 is shown in Table 5. Rating 0 indicates that the active user has not rated jokes 3, 4, 6 and 9. Normalized data of active user in the range 0 to 1 shown in Table 6. Table 7 shows the pheromone value associated with each cluster at time t, similarity of active user profile with the centroid profile of each cluster, computed probability P_i, for i = 1,..,K. Clusters chosen are 2 and 3, since their probability P_i lies in the range $(0.4163\text{-}0.1) \leq P_i \leq 0.4163$.

Step 2: Computing the Rating Quality of Items in Each Chosen Cluster

Rating quality depends on the number of users in the cluster who have rated the item, the individual ratings for the item in the given rating matrix and how close the rating provided by the users is, to each other. The rating quality of the item is computed by the following Eq. 5:

$$Q = \frac{(UB + avg\_rating)}{2 \times UB \times \sqrt{var}} \quad (5)$$

Where:

UB = Upper bound of the ratings
avg_rating = Average rating of the item in the chosen cluster
Var = Variance of the ratings given by individual users for the item in the chosen cluster

As the avg_rating tends to become equal to UB, (UB+avg_rating)/(2*UB) will be close to 1, indicating that users have provided good quality rating. Standard Deviation (SD) is computed as the square root of the variance. A large SD indicates that the ratings are far from the mean and a small SD indicates that they are clustered closely around the mean. The higher value of Q indicates good rating quality and vice versa and guides the pheromone updating step. Table 8 shows the computed rating quality for jokes 3, 4, 6 and 9 which are unrated by active user in chosen clusters 2 and 3.

Step 3: Computing Ratings of Items

Once the quality (Q) of each item which is unrated by active user is computed in the chosen clusters, clusters in which Q for each item lies in the range interval ((highest Q-0.1) ≤ Q ≤ highest Q) are further selected for computing rating instead of only the cluster containing highest Q. Rating of each item is then computed from the selected clusters by computing the weighted average of the ratings using the following Eq. 6:

Table 9: Recommendations generated for active user

Joke	Clusters Chosen	Computed rating
3	2	0.89
4	3	0.14
6	2,3	0.27
9	3	0.14

$$\text{Rating} = \frac{\sum_{cc=1}^{no\_cc} (Q_{cc} \times avg\_rating)}{\sum_{cc=1}^{no\_cc} Q_{cc}} \quad (6)$$

Where:

Q_{cc} = Quality of item in the cluster selected
no_cc = No. of clusters selected
avg_rating = Average rating of the item in the selected cluster

The computed ratings are shown in Table 9. If the number of clusters selected is more than one, then rating is computed as the weighted average, otherwise rating is computed as the average rating of the item in the cluster selected. For joke 2, avg_rating is computed from cluster 2, for jokes 4 and 9, avg_rating is computed from cluster 3 and for joke 6, weighted avg_rating is computed from clusters 2 and 3.

Step 4: Pheromone Updating

The aim of the pheromone updating strategy is to increase the pheromone values associated with good solutions and to decrease those that are associated with bad ones. Usually, this is achieved by decreasing all the pheromone values through pheromone evaporation and by increasing the pheromone levels associated with a chosen set of good solutions. This is analogous to the phenomenon of pheromone evaporation and deposition in real ant colonies. As shown in Eq. 7, the pheromone associated to each cluster is decreased by a small value and pheromone associated to the cluster from which recommendation is generated is increased proportional to the rating quality (ΔQ) of the item. The amount of pheromone associated with each cluster i updated is given as:

$$\text{Pher}_i(t) = (1-\rho) \times \text{Pher}_i(t-1) + \Delta Q \times \text{Pher}_i(t-1) \quad (7)$$

Where:

$$\Delta Q = \begin{cases} \dfrac{Q_{cc}}{\sum_{cc=1}^{no\_cc} Q_{cc}} & \text{if cluster selected } > 1 \\ \dfrac{Q_{cc}}{Q_{cc}+1} & \text{otherwise} \end{cases}$$

Where:

ρ = Pheromone evaporation rate, large value of ρ indicates a fast evaporation and vice versa
Q_{cc} = Rating quality of item in the selected cluster
no_cc = No. of clusters selected

As shown in Table 9, recommendation for joke 3 is generated from cluster 2, joke 4 and 9 are generated from cluster 3 and recommendation for joke 6 is generated from clusters 2 and 3. Therefore,

Table 10: Pheromone computation

	K = 1	K = 2	K = 3
Initial pheromone	0.5000	0.2000	0.3000
Pher value after joke 3 is recommended	0.4643	0.2569	0.2786
Pher value after joke 4 is recommended	0.4263	0.2359	0.3377
Pher value after joke 6 is recommended	0.3325	0.2907	0.3767
Pher value after joke 9 is recommended	0.3026	0.2645	0.4327

ΔQ is computed as $\frac{Q_{cc}}{Q_{cc}+1}$ for jokes 3, 4 and 9 and as $\frac{Q_{cc}}{\sum_{cc=1}^{no\_cc} Q_{cc}}$ for joke 6. This effect on pheromone value is depicted in Table 10.

Step 5: Provide Top-N Recommendations to the Active User

Once the recommendations are generated, top-N recommendations are provided to the active user e.g., For N = 1, joke 3 with rating 0.89 will be recommended, for N = 2, joke 3 and joke 6 with ratings 0.89 and 0.27 will be recommended, respectively and so on.

Step 6: Storing Pheromone Information for Future Recommendations

Pheromone associated to each cluster which is updated based on recommendations made to active user and past recommendations is stored in the database. Pheromone value is higher of the cluster from which recommendations have mostly been generated in past. When a new active user asks for recommendation, similarity of his/her profile with centroid of each cluster is computed to know the taste of the user and combined with the pheromone value computed at time t of the cluster. Clusters that are marked by stronger amount of pheromone have the higher probability of being chosen than those that have weaker amount of pheromone deposit. This helps in improving the quality of recommendations for future users and also gives them new viewpoint of refining their taste.

EXPERIMENTAL STUDY

The experimental study was conducted on three test datasets of varying sizes (small, medium and large) extracted from Jester dataset available online on the site www.ieor.berkeley.edu/~goldberg/jester-data. Our system ARS and its variants were implemented in MATLAB version 7.2 on Microsoft Windows Operating System. The performance was evaluated and compared with traditional collaborative based Recommender System, using Precision, Recall and F1 metrics.

Dataset used in Our Experiments

Our experiments used the Jester dataset (www.ieor.berkeley.edu/~goldberg/jester-data) as test data. Jester is a WWW-based joke recommendation system, developed at University of California, Berkeley. This data has 73,421 user entered numeric ratings for 100 jokes, ranging on real value scale from -10 to +10. The experiments were tested on three datasets extracted from Jester dataset. The datasets are described as follows:

Dataset 1: It consisted of user-item rating matrix of size 10 (users)×10 (jokes) and 5 active users.

Dataset 2: It consisted of user-item rating matrix of size 10 (users)×100 (jokes) and 5 active users.

Dataset 3: It consisted of user-item rating matrix of size 1800 (users)×100 (jokes) and 200 active users.

Evaluation Criteria

Precision and recall are the most popular metrics for evaluating information retrieval systems. For the evaluation of recommender systems, they have been used by various researchers (Billsus and Pazzani, 1998; Basu *et al*., 1998; Sarwar *et al*., 2000a, b). Precision is a measure of exactness or fidelity and recall is a measure of completeness. Precision score of 100% indicates that every recommendation retrieved was relevant. Recall score of 100% indicates that all relevant recommendations were retrieved. Several ways to evaluate precision and recall exists, the most appropriate way is to predict the top-N items for which the user's ratings already exists (Herlocker *et al*., 2004). In this method, a user's ratings are taken and divided into a training set and a test set. Algorithm is then trained on the training set and top N items are predicted from that user's test set. Items that appear in both sets, become members of a special set, called the hit set. This approach is used to evaluate ARS. Recall and precision for top-N recommendation systems can be defined as follows:

Recall is a global measure for the whole dataset. When referring to Recommender Systems, it can be defined as the ratio of the hit set size over the test set size.

$$\text{Recall} = \frac{\text{Size of hit set}}{\text{Size of test set}} = \frac{|\text{test} \cap \text{top} - \text{N}|}{|\text{test}|} \tag{8}$$

Precision, when referring to recommender systems, can be defined as the ratio of hit set size over the top-N set size. It gives the average quality of an individual recommendation.

$$\text{Precision} = \frac{\text{Size of hit set}}{\text{Size of top N set}} = \frac{|\text{test} \cap \text{top} - \text{N}|}{\text{N}} \tag{9}$$

These two measures are clearly conflicting in nature. If the number of recommendations (N) produced increases, then the value of recall increases, while at the same time precision is decreased. But, since both recall and precision are important in evaluating the performance of a system that generates top-N recommendations, they can be combined with equal weights, to get a single metric, the F_1 metric. It consists of weighted combination of precision and recall. The general formula for F-measure (for non-negative real, α) is:

$$F_\alpha = \frac{(1+\alpha^2) \times \text{Precision} \times \text{Recall}}{\alpha^2 \times \text{Precision} + \text{Recall}} \tag{10}$$

When $\alpha = 1$, it is known as F_1 measure and represents the weighted harmonic mean of precision and recall giving equal weights to them:

$$F_1 = \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \times 2 \tag{11}$$

Higher values of F_1 indicate a more balanced combination between recall and precision.

Experimental Results

Ant Recommender System (ARS) was implemented in MATLAB version 7.2 on Microsoft Windows Operating System. Ant based clustering algorithm was implemented to cluster the user-item rating matrix as the initial step. Experiments were conducted on datasets 1, 2 and 3 by varying the values of parameters R (number of ants), L (No. of solutions on which local search is performed), maximum number of iterations, Q0 (threshold value) and keeping ρ (pheromone evaporation rate) = 0.01 constant. The best values for the parameters experimentally determined for K

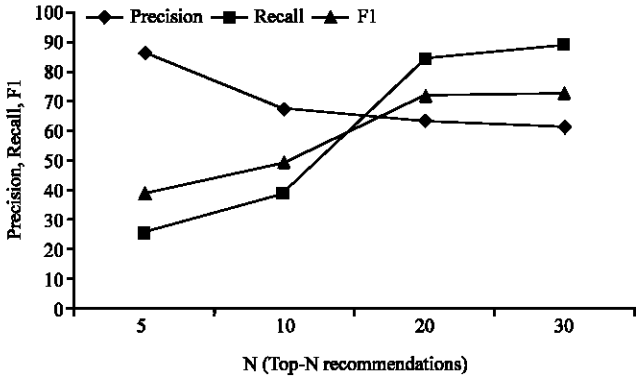

Fig. 2: ARS: Precision, recall, F1 values for K = 10

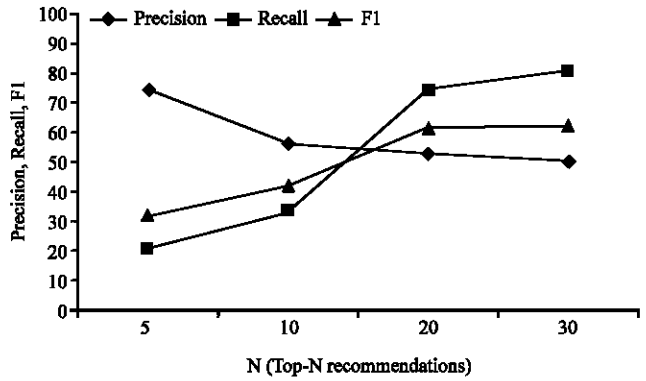

Fig. 3: CF based RS: Precision, recall, F1 values for K = 10

(No. of clusters) = 10, 20 and 30 when dataset 3 was used were as follows: R = 2000, L = 20%, maximum number of iterations = 10000 and Q0 = 0.70.

In addition to ARS, traditional collaborative filtering based recommender system was also implemented to test the performance of our system. By varying the number of clusters, a top-N recommendation list was evaluated using recall, precision and F_1 measure via., Eq. 8, 9 and 11, respectively for both the systems. Figure 2, 4 and 6 show results of evaluation metrics for ARS and Fig. 3, 5 and 7 for traditional collaborative filtering based recommender system when number of clusters (K) = 10, 20 and 30. Table 11 and 12 show the results of evaluation metrics when dataset 3 was used which indicates that ARS performed better on dataset 3 as compared to traditional collaborative filtering based recommender system. Recommendations generated using pheromone strategy of ants were better as compared to recommendations generated based only on similarity measure.

It can be seen in Fig. 2-7 that, precision dropped smoothly as the number of recommendations (N) increased. This is expected as the average quality of the recommendations made decreases as the number of recommendation to be made increases. On the other hand, recall increased as the value of N increased from 5 to 30. F_1 measure indicates that the best combination of precision and recall is achieved for high values of N (N = 20 and N = 30).

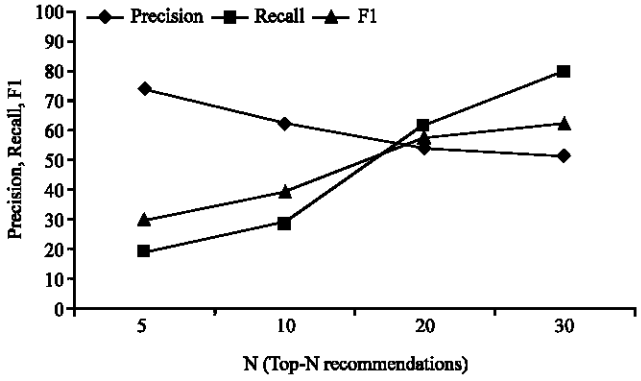

Fig. 4: ARS: Precision, Recall, F1 values for K = 20

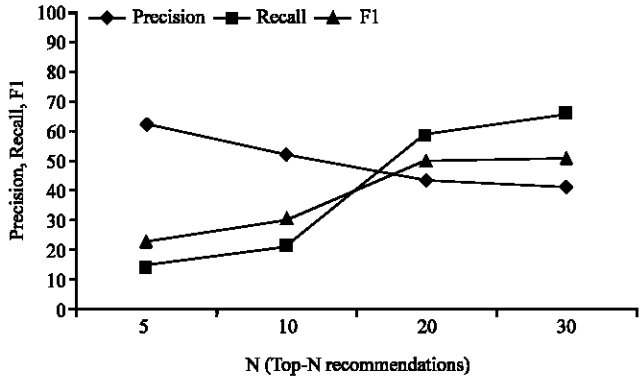

Fig. 5: CF based RS: Precision, Recall, F1 values for K = 20

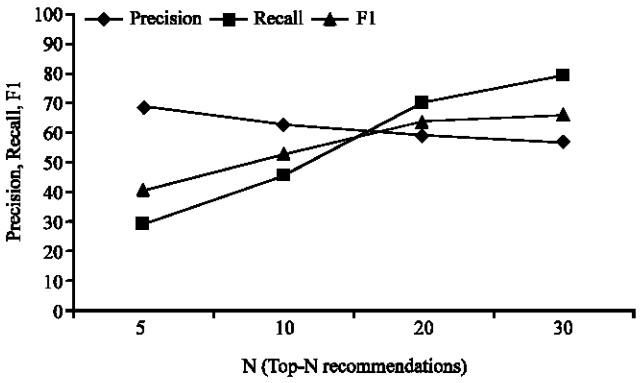

Fig. 6: ARS: Precision, Recall, F1 values for K = 30

Other variants to our approach were also implemented viz., (a) clustering the user-item rating matrix using k-means algorithm in phase I and applying pheromone updating strategy of ants while generating recommendations in phase II. (b) Clustering the user-item rating matrix using ant based

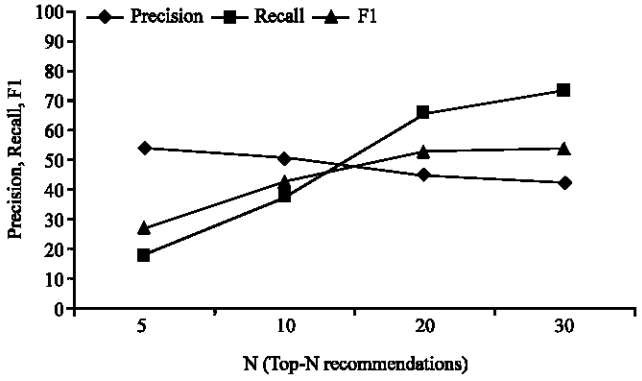

Fig. 7: CF based RS: Precision, Recall, F1 values for K = 30

Table 11: ARS: Ant based clustering technique used in phase I and recommendations generated using pheromone strategy of ants in phase II

	No. of clusters =10			No. of clusters =20			No. of clusters =30		
N	P	R	F1	P	R	F1	P	R	F1
5	85.71	24.68	38.32	73.65	18.77	29.91	68.52	28.74	40.49
10	66.68	38.39	48.72	62.5	28.51	39.15	62.85	45.57	52.83
20	62.86	84.00	71.90	54.32	61.55	57.70	58.99	69.27	63.71
30	61.00	88.50	72.22	51.43	79.52	62.46	56.89	79.22	66.22

P: Precision in %, R: Recall in %, F1: F1 metric

Table 12: Traditional collaborative filtering based recommender system: k-means clustering technique used in phase I and recommendations generated based only on similarity measure in phase II

	No. of clusters =10			No. of clusters =20			No. of clusters =30		
N	P	R	F1	P	R	F1	P	R	F1
5	74.42	20.12	31.67	62.25	14.00	22.85	54.25	17.73	26.72
10	56.12	33.34	41.82	51.98	20.90	29.81	50.16	36.98	42.57
20	52.89	74.37	61.81	43.33	58.55	49.80	44.27	65.48	52.82
30	50.27	81.01	62.04	41.30	65.52	50.66	42.21	72.99	53.48

P: Precision in %, R: Recall in %, F1: F1 metric

algorithm in phase I and generating recommendations based only on similarity measure in phase II. It was observed that on dataset 1, recommendations generated by traditional collaborative filtering based recommender system and variant (a) were approximately same but better than ARS. On datasets 2 and 3, recommendations generated by variant (a) were better than traditional collaborative filtering based recommender system. On datasets 2 and 3, ARS performed better as compared to traditional collaborative filtering based recommender system. From the results obtained, we can conclude that ant based technique works well for large dataset.

DISCUSSION

An approach Ant Recommender System (ARS) based on biological metaphor of ant colonies for generating Top-N recommendations is proposed and tested on Jester dataset available online. Improving the quality of recommendations is one of the fundamental challenges for recommender systems. Pheromone updating strategy of ants serve as an added dimension to traditional collaborative filtering based approach for generating good quality recommendations. In our method, pheromone associated to each cluster is decreased by a small value and pheromone associated to the cluster from

which recommendation is generated is increased proportional to the rating quality (ΔQ) of jokes. Rating quality (Q) of each joke is computed taking three factors into account viz. highest rating, average rating and variance computation of ratings given by users. The higher value of Q indicates good rating quality and vice versa and guides the pheromone updating step. The results obtained shows that it is promising to apply pheromone updating strategy of ant colony metaphor when the dataset is large i.e., millions of users interact indirectly.

It is difficult for us to compare our results with the work of other researchers as most of them have used Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE) as the evaluation metrics. Mean absolute error and NMAE represent the absolute difference between real and predicted values. These measures characterize the accuracy of prediction, but since, our work focuses on quality and accuracy of recommendations; we have used the widely accepted metrics from information retrieval namely precision, recall and F1 measure. For the evaluation of recommender systems, these measures have been used by various researchers (Billsus and Pazzani, 1998; Basu *et al*., 1998; Sarwar *et al*., 2000a, b; Symeonidis *et al*., 2008) for the datasets collected from each movie collaborative filtering service, Movielens movie recommendation site and web-purchasing transaction of a large E-Commerce company. Researchers have also applied Swarm Intelligence techniques in recommender systems to intermediate steps but not evaluated results using precision, recall and F1 measure. Examples include learning personal preferences of users and providing tailored suggestions (Ujjin and Bentley, 2003), retrieval step of the recommendation cycle (Lorenzi *et al*., 2005) and ant colony metaphor is used for selecting the most optimal path in the web-based system user interface hybrid recommendation method (Sobecki, 2008). Some researchers, who have taken Jester dataset for experimental study, have not used precision, recall and F1 measure. Other researchers who have used precision, recall and F1 measure, have not taken Jester dataset for experimental study and their tables or graphs showing the results after evaluation are concise. This was the reason; we implemented traditional collaborative filtering based recommender system ourselves and compared the results in Table 11 and 12.

As shown from Fig. 2-7, precision dropped smoothly as the number of recommendations (N) increased. This is expected as the average quality of the recommendations made decreases as the number of recommendation to be made increases. On the other hand, recall increased as the value of N increased. F1 measure indicates that the best combination of precision and recall is achieved for high values of N (N = 20 and N = 30). This effect is summarized in Table 11 and 12. Results depict that precision is increased by approximately 10%, recall is increased in the range 3 to 10% and F1 metric is increased in the range 7 to 10%, while generating recommendations using ARS as compared to traditional collaborative filtering based recommender system.

CONCLUSION

Recommender Systems are important tools to overcome the information overload, by sifting through the large set of data and recommending information relevant to the user. They base their operation on user ratings/preferences over a collection of items. In ARS, opinions from users collected in the form of user-item rating matrix are clustered using ant based clustering algorithm. The algorithm optimally clusters rating matrix into K clusters. In traditional CF based recommender systems, similarity is normally the only heuristic used during the recommendation process whereas in ARS, similarity measure is combined with the pheromone updating phenomenon known from ant algorithms. The pheromone information helps in exploration of other clusters which have similarity closer to the taste of active user and provides him with good set of alternative recommendations. Also, we have tried to improve the quality of recommendations based on the pheromone density and pheromone updating strategy, which is one of the fundamental challenges for recommender systems. The performance of the proposed system is experimentally investigated using Jester dataset which is available on the

website of University of California, Berkeley. The ARS is compared with traditional collaborative filtering based recommender system and the results show that it works better for large dataset.

REFERENCES

Adomavicius, G. and A. Tuzhilin, 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowledge Data Eng., 17: 734-749.

Basu, C., H. Hirsh and W.W. Cohen, 1998. Recommendation as classification: Using social and content-based information in recommendation. Proceedings of the 15th National Conference on Artificial Intelligence, Jul. 31, AAAI Press, Madison, Wisconsin, pp: 714-720.

Billsus, D. and M.J. Pazzani, 1998. Learning collaborative information filters. Proceedings of the 15th International Conference on Machine Learning, Madison, Jul. 24-27, Morgan Kaufman San Francisco, CA., pp: 46-54.

Blum, C., 2005. Ant colony optimization: Introduction and recent trends. Physics Life Rev. J., 2: 353-373.

Dorigo, M., V. Maniezzo and A. Colorni, 1996. The ant system: Optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybernet., 26: 29-41.

Dorigo, M. and T. Stutzle, 2004. Ant Colony Optimization. MIT Press, Cambridge, MA, ISBN-10: 0-262-04219-3.

Goldberg, D., D. Nichols, B.M. Oki and D. Terry, 1992. Using collaborative filtering to weave an information tapestry. Commun. ACM., 35: 61-70.

Goldberg, K., T. Roeder, D. Gupta and C. Perkins, 2001. Eigentaste: A constant time collaborative filtering algorithm. Inform. Retrieval J., 4: 133-151.

Handl, J. and B. Meyer, 2007. Ant-based and swarm-based clustering. Swarm Int. J., 1: 95-113.

Herlocker, J.L., J.A. Konstan, A. Borchers and J. Riedl, 1999. An algorithmic framework for performing collaborative filtering. Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Aug. 15-19, Berkeley, California, United States, ACM Press, pp: 230-237.

Herlocker, J.L., J.A. Konstan and J. Riedl, 2000. Explaining collaborative filtering recommendations. Proceedings of the ACM Conference on Computer Supported Cooperative Work, Philadelphia, Pennsylvania, United States, 2000, ACM Press, USA., pp: 241-250.

Herlocker, J.L., J.A. Konstan, L.G. Terveen and J. Reidl, 2004. Evaluating collaborative filtering recommender systems. ACM Trans. Infrom. Syst., 22: 5-53.

Hill, W., L. Stead, M. Rosenstein and G. Furnas, 1995. Recommending and evaluating choices in a virtual community of use. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Denver, May 7-11, ACM Press/Addison-Wesley Publishing Co., Colorado, United States, pp: 194-201.

Huang, Z. and D.D. Zeng, 2005. Why does collaborative filtering work? Recommendation model validation and selection by analyzing bipartite random graphs. Proceedings of the Workshop of Information Technologies and Systems, Oct. 17, Las Vegas, NV., pp: 1-6.

Kekeç, G., N. Yumu°ak and N. Çelebi, 2006. Data mining and clustering with ant colony optimization. Proceedings of 5th International Symposium on Intelligent Manufacturing Systems, May 29-31, Sakarya University, Department of Industrial Engineering, pp: 1178-1190.

Konstan, J.A., B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon and J. Riedl, 1997. Group lens: Applying collaborative filtering to use net news. Comm. ACM, 40: 77-87.

Kuo, R.J., J.L. Liao and C. Tu, 2005. Integration of ART2 neural network and genetic K-means algorithm for analyzing web browsing paths in electronic commerce. Dec. Support Syst., 40: 355-374.

Lorenzi, F., D.S. dos Santos and A.L.C. Bazzan, 2005. Case-based recommender systems inspired by social insects. Proceedings of XXV Congresso da Sociedate Braseilera de Computacao, 2005, Sao Leopolda, pp: 752-760.

Resnick, P., N. Lakovou, M. Sushak, P. Bergstrom and J. Riedl, 1994. Group lens: An open architecture for collaborative filtering of Netnews. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, Oct. 22-26, Chapel Hill, North Carolina, United States, ACM Press, pp: 175-186.

Rich E., 1979. User modeling via stereotypes. Cognitive Sci. J., 3: 329-354.

Saatchi, S. and C.C. Hung, 2005. Hybridization of the Ant Colony Optimization with the K-Means Algorithm for Clustering. In: Scandinavian Conference on Image Analysis, Kalviainen, H. *et al*. (Eds.). LNCS 3540, Springer-Verlag, Berlin, Heidelberg, ISBN: 978-3-540-26320-3, pp: 511-520.

Sarwar, B., G. Karypis, J. Konstan and J. Riedl, 2000a. Analysis of recommendation algorithms for e-commerce. Proceedings of the 2nd ACM Conference on Electronic Commerce, Oct. 17-20, ACM Press, Minneapolis, MN, USA., pp: 158-167.

Sarwar, B.M., G. Karypis, J.A. Konstan and J. Reidl, 2000b. Application of dimensionality reduction in Recommender system: A case study. ACM WebKDD 2000 Web Mining for E-Commerce Workshop.

Sarwar, B., G. Karypis, J. Konstan and J. Reidl, 2001. Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th International World Wide Web Conference, May 1-5, Hong Kong, pp: 285-295.

Schafer, J.B., D. Frankowski, J. Herlocker and S. Sen, 2007. Collaborative Filtering Recommender Systems. In: The Adaptive Web, Brusilovsky, P., A. Kobsa and W. Nejdl (Eds.). LNCS 4321, Springer-Verlag, Berlin, Heidelberg, ISBN: 978-3-540-72078-2, pp: 291-324.

Shardanand, U. and P. Maes, 1995. Social information filtering: Algorithms for automating Word of Mouth. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, May 7-11, Denver, Colorado, United States, pp: 210-217.

Shelokar, P.S., V.K. Jayaraman and B.D. Kulkarni, 2004. An ant colony approach for clustering. Anal. Chim. Acta, 509: 187-195.

Sobecki, J., 2008. Web-Based System User Interface Hybrid Recommendation Using Ant Colony Metaphor. In: Knowledge-Based Intelligent Information and Engineering Systems, Apolloni, B. *et al*. (Eds.). LNCS 4694, Springer-Verlag, Berlin, Heidelberg, ISBN: 978-3-540-74828-1, pp: 1033-1040.

Symeonidis, P., A. Nanopoulos, A.N. Papadopoulos and Y. Manolopoulos, 2008. Collaborative recommender systems: Combining effectiveness and efficiency. Expert Syst. Appl., 34: 2995-3013.

Terveen, L., W. Hill, B. Amento, D. McDonald and J. Creter, 1997. PHOAKS: A system for sharing recommendations. Commun. ACM, 40: 59-62.

Ujjin, S. and P.J. Bentley, 2003. Particle swarm optimization recommender system. Proceedings of the 2003 IEEE Swarm Intelligence Symposium-SIS '03, April 24-26, Department of Computer Science, University Coll. London, UK., pp: 124-131.

Vrahatis, M.N., B. Boutsinas, P. Alevizos and G. Pavlides, 2002. The new K-windows algorithm for improving the K-means clustering algorithm. J. Complexity, 18: 375-391.

Walker, A., M.M. Recker, K. Lawless and D. Wiley, 2004. Collaborative information filtering: A review and an educational application. Int. J. Artifi. Intell. Educ., 14: 3-28.