Optimizing the Number of Neighbors in Trust Based Recommender Systems

Harmeet Kaur[1] and Deepali Jain[2]

[1]**Department of Computer Science, Hans Raj College, University of Delhi**
Delhi - 110007, India

[2]**Department of Computer Science, Hans Raj College, University of Delhi**
Delhi - 110007, India

Abstract

Users in trust based recommender systems seek recommendations from their directly trusted parties known as neighbors or from neighbors of neighbors and so on. This paper proposes an appropriate range ([min_{rec}… max_{rec}]) providing minimum and maximum number of recommenders that one should have in his close communication. More neighbors than the maximum number of neighbors (max_{rec}) suggested by the range does not improve the quality of recommendations but requires more time and computations in accumulation of extra recommendations. Lesser number of neighbors than min_{rec} may compromise the quality of recommendations thus requiring seeking recommendations transitively. This again involves time and computation in propagating the query through chain of neighbors and getting the responses. A method to maintain number of neighbors within this range is also proposed. Experiments were conducted on real datasets to discover the most appropriate number of neighbors that an agent should keep.

Keywords: *Trust, Recommender System, Optimal number of recommenders*

1. Introduction

With the overwhelming amount of information available on the World Wide Web, it is tremendously complicated for users to pick out the best possible option for them. Information overload has become an increasingly common problem in today's large scale internet applications where users are dealing with very large amounts of data that can become time consuming to analyze [21]. Thus it is important to have tools to help users to select the relevant part of online information. A popular way to address this matter is to use recommender systems. Recommender systems are heavily used in e-commerce to provide users with high quality, personalized recommendations to help them find satisfactory items (e.g. books, movies, news, music, etc.) among a huge number of available choices[7].

In general, recommender systems suggest items by matching the attributes of an item to the profile of the user (content-based recommendation), or by correlating the profile of the user (or items selected by him) with others in the system (collaborative filtering) [23]. However, these systems do not take account of how people seek recommendations from their social networks of known individuals. Since trust is a vital ingredient of any successful interaction between individuals, among organizations and/or in society at large [2], thus trust should be incorporated in recommender systems. Trust-enhanced recommender systems refine the classical recommendation techniques, by making use of trust relationships between users in a network [5]. Trust based recommender systems provide recommendations by mining the trust network referred to as Web of Trust (WoT) among its users. These trustworthy connections among users commonly take the form of weighted trust assertions, indicating how much one user trusts another. Therefore trust based recommender systems incorporate trust network known as Web of Trust (WoT) where each user is represented by his agent and these agents collaborate on the basis of trust.

This paper determines the numeric values for min_{rec} (minimum number of neighbors) and max_{rec} (maximum number of neighbors) for the number of neighbors that an agent should maintain in WoT. Significance of sustaining number of neighbors from this range can be understood by considering two cases. First case is where an agent in WoT is connected to more than max_{rec} neighbors then that agent will have extra recommendations which may include repeated and irrelevant recommendations. This not only consumes agent’s time but computations also, to accumulate redundant recommendations. Second case is where an agent has less than min_{rec} neighbors, this result in losing valuable suggestions available with other agents which again is not a good option. A solution to the problem discussed in first case lies in removing extra number of recommenders from neighborhood where as to deal with the problem mentioned in second scenario, an agent needs to increase the number of agents in its neighborhood.

This paper also presents the procedures of increasing as well as decreasing the number of agents in one's neighborhood so as to maintain the number of neighbors in the range [min_{rec}… max_{rec}]. Experiments have been conducted to verify the validity of this range.

Main contributions of the paper are summarized as follows:

(1) This paper proposes an appropriate range for the number of neighbors that an agent should maintain in its neighborhood and experiments have been conducted on real data set to demonstrate the validity of range [min_{rec} …max_{rec}].

(2) Provides a method for expansion of neighborhood through which an agent can expand its neighborhood and include some more good recommenders in its direct approach to get more and better suggestions in reduced amount of time and computations.

(3) Presents a technique of contracting of neighborhood to reduce number of neighbors from current neighborhood of an agent and thereby reducing time and computations involved in filtering preferred recommendations from a large set of redundant recommendations.

Organization of this paper is as follows: related work is discussed in section 2. Some preliminary details of WoT are given in section 3. The proposed models of neighborhood expansion and neighborhood contraction are explained in section 4 and 5 respectively. Experimentation and results hence obtained are reported in section 6. Finally, Section 7 concludes the paper and presents some directions for future work.

2. Related Study

Trust has been extensively studied in recommender systems and successfully employed to improve classical recommendation techniques significantly. In literature, there are several algorithms for calculating trust on the web of trust network, and they use different operators and methods to infer trust.

O'Donovan and Smyth proposed a profile- and item-based recommendation that takes into consideration both the similarities among users and the trustworthiness of recommendation histories. Their trust metrics compute the percentage of correct recommendations that the user has contributed. Trust is built up between users x and y, by measuring trust of consumer (user) on producer (recommender) y as the percentage of the correct recommendations received by x from y [16].

In [15] Golbeck introduces the Tidal Trust algorithm to estimate trust values between actor pairs in a social network. One agent infers trust rating for another by using a weighted average over all neighbors.

Massa and Avesani studied the trust-aware recommender systems [19]. Their work replaces the similarity finding process with the use of a trust metric, which is able to propagate trust over the trust network and to estimate trust weight. They propose Mole Trust which performs depth-first search, to propagate and infer trust in the trust network.

Bedi et al. in [18] proposed a trust-based recommender system for the Semantic Web; this system runs on a server with the knowledge distributed over the network in the form of ontologies, and uses the Web of trust to generate the recommendations.

Jamali and Ester [17] design the Trust-Walker approach to randomly select neighbors in the trust network formed by users and their trusted neighbors. Trust information of the selected neighbors is combined with an item-based technique to predict item ratings.

In [12] paper selection of trustworthy recommenders was done on the basis of entropy between the users. Authors have developed entropy based computational model which operates at two levels and recommenders were generated by monitoring entropy between similar users.

The model presented in [6] consists of agents, objects, and agent's profiles. In this model whenever a source agent wants to rate a particular item it asks its neighbors and its neighbors in turn pass on a query to their neighbors if they cannot provide a rating themselves. In order to generate the transitive trust from source agent to sink agent they have used the multiplicative approach and multiply the trust values along the path between the source and sink agent.

Most of the existing trust based recommender systems follow one of the following methods of assembling recommendations:

1. Getting recommendations from directly trusted associates only, i.e. only from those agents in WoT which are in the direct link of the recommendation seekers agent known as neighbors. [18]
2. Getting recommendations transitively by propagating the query through the chain of connections towards user's neighbors of neighbors and so on. [1, 3, 4, 6, 8, 9, 10, 15, 19,20, 24]

The above mentioned techniques have not considered the optimal number of recommenders while generating recommendations. However some researchers have proposed to obtain recommendations from topmost k recommenders only, but they have not provided a numeric value for k or some minimum or maximum number of recommenders. Thus this paper proposes a range providing minimum and maximum number of neighbors an agent should possess in order to retrieve useful and complete recommendations in least amount of time and effort.

3. Web of Trust

Trust based recommendation systems usually construct a trust network called Web of Trust (WoT) where nodes are users and edges represent trust between two users. It is a virtual community of agents where agents interact and cooperate with each other in order to find valuable information for their human users [11]. The goal of a trust based recommendation system is to generate personalized recommendations by aggregating the opinions of users in their trust network [22].In WoT each agent is connected to a number of agents in web of trust which forms its neighborhood.

In WoT, trust is initialized on the basis of ability of an agent to give good recommendations and is updated using actual interactions. Boolean expressions such as trust or no trust is not appropriate for user users in social network. In real life scenarios an element of vagueness is always involved while assigning trust to a known social contact or a friend. Thus fuzzy logic is very well-suited to represent such natural language labels which represent vague intervals rather than exact values. Instead of assigning trust in crisp terms one tends to assign it in the range of 0 to 1 where 0 defines no trust and 1 symbolizes total trust. This paper uses degree of trust (where degree ranges from 0 to 1) to represent trust between two agents.

Web of Trust (WoT) can be viewed as a directed graph where:

- Agents are represented by nodes of the graph.
- Directed link from source vertex to the target vertex represents the fact that agent associated with source trusts agent linked to the target vertex.
- Weights of edges of the directed graph are annotated with the degree of trust from source to target, where this degree ranges from 0 to 1(taking trust as fuzzy value)

Figure 1 depicts an example web of trust where nodes symbolizing agents are connected through directed edges. Presence of the directed edge from agent a_i to agent a_j furnishes the information that a_i trusts a_j and the weight of this edge that is t_{ij} is the degree of trust from a_i to a_j which stands for the extent to which a_i trusts a_j to give good and useful recommendations.

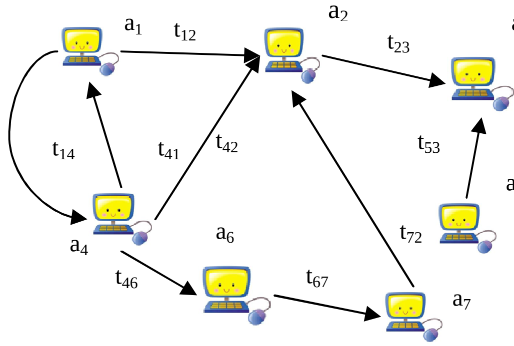

Fig. 1 Web of Trust

If there is a direct trust path in Web of Trust from agent a_i to agent a_j then agent a_i can directly take a_j's suggestions into consideration. Here comes the significance of number of agents directly connected to an agent. If sufficient numbers of agents are in direct contact with the user agent then there is no need to propagate user query and finding trust on non adjacent agents transitively which saves time as well as computations, therefore in the situation less than required number of agents are connected to source, it can invoke process of expansion of neighborhood and include some more trustworthy agents in its direct association.

In the case where source is connected to large number of agents then again it will result in wastage of time and effort in accumulation of redundant responses, thus it has to restrict the response accumulation process by removing lesser trusted parties from its neighborhood.

There is no role of centralized authority in web of trust to maintain data repository and performing calculations to generate and process recommendations. Each agent is responsible for maintaining its data and carry out computations to generate and aggregate recommendations.

4. Expansion of neighborhood

In a scenario where source (recommendation seeker) has less than min_{rec} number of agents in its neighborhood, it will have to propagate its request towards its neighbors of neighbors and so on until its query is satisfied. This results in involvement of time and computations in reaching suitable and trustworthy distant agents and fetching results from those agents in addition to finding trust on those distantagents. In order to avoid the additional load of accumulating recommendations transitively each time the query propagates, an agent should increase members in its neighborhood by having good recommenders as neighbors. For the purpose of including any new agent in the neighborhood source agent will have to estimate its trustworthiness as well, thus one would require a procedure to calculate trustworthiness of newly added agents only once so that in future their recommendations could be taken without wasting much time and effort. This paper presents an algorithm of expanding neighborhood to add more agents in source's neighborhood where calculation of degree of trust for distant agents happens once and later on source can fetch their advice directly.

When source agent say a_i wishes to include some more agents in its neighborhood, it carries out the process of expansion of neighborhood. Procedure of expansion of neighborhood involves two main steps:

1. Propagation of the request from source towards its neighbors to suggest trustworthy agents that can be added into the source's neighborhood.
2. Accumulation of responses from neighbors and computing trustworthiness of newly suggested agents.

4.1 Request Propagation

As a part of request propagation procedure, the source prepares a request with the following 5-tuple query

$\langle$request_id, trust_threshold_neighbor, k, item_list, liking_list$\rangle$

where

- **request _id** is the unique identification number of the request,
- **trust_threshold_neighbor** defines the minimum value of trust in an immediate neighbor so that the request can be propagated to that neighbor,
- source's neighbors searches their list of acquaintances and report its **k**th most trusted neighbor. Initially **k** is set to be 1. This parameter **k** will help in finding some more trustworthy agents in subsequent invocation of expansion process, if required, as each time agent a_i calls this process its neighbors will report different agents in decreasing order of trustworthiness,
- In order to be included in source's neighborhood, agents suggested by source's neighbors must review 'm' items provided in the **item_list** prepared by the source. These agents will be inquired about their likes and dislikes for these items which is then used to find their similarity with the source and to judge their trustworthiness,
- liking_list is a list of m entries where each entry gets filled by either 1 or 0 on completion of algorithm1,mth entry of this list corresponds to mth item in the item_list.

Source agent a_i prepares the request and finds the trust t_{ij} on all its neighboring agents a_j. For all the neighboring agents a_j such that t_{ij}>trust_threshold_neighbor, a_i sends a request to find trustworthy agents. When a neighboring agent a_j of a_i receives a request in the form of a 5-tuple from the source, it undertakes steps as outlined in Algorithm 1.

Algorithm 1: Request Propagation

1. a_j searches its list of neighbors
2. from all the neighboring agents of a_j, it selects its kth most trusted neighbor say a_{jk}
   2.1 a_j retrieves and collects likes and dislikes of a_{jk} about items present in item_list
3. a_j populates liking_list of a_{jk} such that

$$\text{liking\_list}[x]=\begin{cases}1 & \text{if } a_{jk} \text{ likes } x^{th} \text{ item in the item list} \\ 0 & \text{otherwise}\end{cases} \quad (1)$$

4. Send response as <a_j, a_{jk}, liking_list> to the sender of the request, a_i

4.2 Response Accumulation

A response is a tuple of the form < sender, nominee_agent, liking_list> where

sender is the one who is sending the response towards the source,

nominee_agentis the agent recommended by sender, and

liking_list is the populated liking_list of the suggested agent.

Algorithm 2 outlines the steps taken by the source when it receives all such responses from all its neighbors a_j, where every response is of the form<a_j, a_{jk}, liking_list>.

Algorithm 2: Response Accumulation

1. a_i prepares n x 1 column matrix N of nominee agents

$$N=\begin{bmatrix} n_1 \\ \vdots \\ n_n \end{bmatrix} \quad (2)$$

Here, n_p is the pth nominee,

2. a_i prepares n x 1 column matrix T

$$T=\begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix} \quad (3)$$

Here, t_p(equivalent to t_{ij} of Algorithm 1) is trust between source and its neighbor which has recommended p^{th} nominee,

3. a_i arranges CM (choice matrix) which is the matrix of order m x 1 that represents the source's likes and dislikes about the m items in the item_list

$$\mathrm{CM} = \begin{bmatrix} \mathrm{cm}_1 \\ \vdots \\ cm_m \end{bmatrix} \quad (4)$$

Here

$$\mathrm{cm}_q = \begin{cases} 1 & \text{if source agent likes item q} \\ 0 & \text{if source agent dislikes item q} \end{cases}$$

4. DOI is also a matrix of order m x 1 that represents the importance source associates with the m items in the item_list.

$$\mathrm{DOI} = \begin{bmatrix} \mathrm{doi}_1 \\ \vdots \\ doi_m \end{bmatrix} \quad (5)$$

Here $0 \le \mathrm{doi}_q \le 1$ and $\sum_{q=1}^{m} \mathrm{doi}_q = 1$

5. a_i carries out steps mentioned in algorithm 3 given in Section 4.3 which calculates DOT (Degree of Trust) matrix. DOT is a matrix of order n x 1 having the calculated values of degree of trust on nominee agents.
6. if DOT_p>trust_threshold_new_agent (6)
   6.1 include p^{th} nominee in final_agents_list
7. if number of agents in final_agents_list<min_{rec} then
   7.1 k= k+1 (7)
   7.2 source agent repeatsalgorithm1 and algorithm 2.

Algorithm 1and 2 outline the steps taken when source agent a_i wants to expand its neighborhood by adding some more trustworthy neighbors in its close association. The agent a_i asks its immediate neighbors about some credible nominee to become its neighbor. Since trust decays with the increase of number of hops along social trust pathand trust decay is commonly agreed upon,for people tend to trust individuals trusted by immediate friends more than individuals trusted by friends of friends and so on [14]. Thus the process of inquiring about some trustworthy neighbors from immediate neighbors takes trust decay into account as query propagation is restricted up to one level away from source. Neighboring agents respond by giving the names of its trustworthy agent that becomes a nominee. Using algorithm 3, source computes degree of trust on the nominee agents and if this computed degree of trust is greater than trust_threshold_new_agent, source will include that nominee in its neighborhood. To include more agents, if required, in the neighborhood, source will again repeat algorithms 1and 2 with k = k+1.

4.3 Computation of trustworthiness of new agents

In order to ascertain degree of trust on agents recommended by neighbors, source agent calls algorithm 2 in which it prepares nominee-item (NI) matrix of order n x m, having n rows for n nominee agents and m columns for m items. Algorithm 2 populates this nominee-item matrix by accumulating nominee's likes and dislikes about m items. Corresponding to each liking for j^{th} item by i^{th} agent, a 1 is inserted in $NI[i, j]^{th}$ position of NI matrix, where as for dislike a 0 is inserted.

Algorithm 3: Computation of degree of trust on nominee agents by the source agent

1. Prepare nominee-item (NI) matrix of order n x m, using matrix N and n liking_lists

$$\mathrm{NI} = \begin{bmatrix} ni_{11} & \cdots & ni_{1m} \\ \vdots & \ddots & \vdots \\ ni_{n1} & \cdots & ni_{nm} \end{bmatrix} \quad (8)$$

Here, n is the number of nominees,
m is the number of items, and

$$\mathrm{ni}_{ij} = \begin{cases} 1 & \text{if nominee i likes item j} \\ 0 & \text{if nominee i dislikes item j} \end{cases}$$

2. Compute weighted NI (WNI) matrix from NI and DOI of order n x m as follows:

$$\mathrm{WNI} = \begin{bmatrix} ni_{11} \times doi_1 & \cdots & ni_{1m} \times doi_m \\ \vdots & \ddots & \vdots \\ ni_{n1} \times doi_1 & \cdots & ni_{nm} \times doi_m \end{bmatrix} \quad (9)$$

3. Compute similarity matrix SC as product of two matrices WNI and CM resulting in SC matrix to be of order n x 1.
   SC = WNI × CM (10)
4. For each nominee a_p , compute the final trust on a_p as follows
   4.1 DOT = α× SC+ β × T (11)
   Here

$$\mathrm{DOT} = \begin{bmatrix} \mathrm{dot}_1 \\ \vdots \\ dot_n \end{bmatrix} \quad (12)$$

DOT is then n x 1 matrix where $dot_p$is computed trust between source and p^{th} nominee,
SC is the similarity matrix,
T is the is n x 1 matrix, t_p is trust between source and its neighbor which has recommended p^{th} nominee,

α is the weight of similarity parameter,
β is the weight of trust parameter, and
α + β = 1.

5. Return matrix DOT as result which contains computed degree of trust on nominees

Algorithm 3 is used to compute degree of trust for agents where different weights are provided to various items. These weights are nothing but their degree of importance assigned by the source. Correlation between similarity and trust has already been proved by [4, 15, 19]. Algorithm 3 uses similarity matrix SC which furnishes the information about similarity between source and agents recommended by source's neighbors by comparing their likes and dislikes for items in item_list. Finally degree of trust from source to newly proposed agents is computed using equation (11) where parameters α and β controls relative importance of similarity index and trust value respectively. If the source is interested in looking similar agents to be included in its neighborhood then more weight should be assigned to α and on the contrary if source would like to give more weight to the path in WoT from where it is coming then α<β.

5. Contraction of neighborhood

Consider the situation where source is in direct communication with a large number of agents, more than max_{rec}, then it leads to wastage of time and effort in gathering of superfluous responses from extra neighbors, thus source has to adopt the procedure of contraction of neighborhood given in algorithm 4to remove lesser trusted parties from its neighborhood.

Algorithm 4: Neighborhood Contraction

1. a_i prepares n x 1 column matrix N of neighboring agents

$$N=\begin{bmatrix} n_1 \\ \vdots \\ n_n \end{bmatrix} \quad (13)$$

Here, n_p is the p^{th} neighbor,

2. a_i prepares n x 1 column matrix T

$$T=\begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix} \quad (14)$$

Here, t_p is trust between source and its p^{th} neighbor,

3. a_i retrieves and collects likes and dislikes of a_p about items present in item_list, for all p

4. a_i populates liking_list of its p^{th} neighbor such that

$$\text{liking\_list}[x]=\begin{cases} 1 & \text{if } a_p \text{ likes } x^{th} \text{ item in the item list} \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

5. a_i arranges CM which is the matrix of order m x 1 that represents the source's likes and dislikes about the m items in the item_list

$$CM=\begin{bmatrix} cm_1 \\ \vdots \\ cm_m \end{bmatrix} \quad (16)$$

$$\text{Here}\, cm_q=\begin{cases} 1 & \text{if source agent likes item q} \\ 0 & \text{if source agent dislikes item q} \end{cases}$$

6. DOI is also a matrix of order m x 1 that represents the importance source associates with the m items in the item_list.

$$DOI=\begin{bmatrix} doi_1 \\ \vdots \\ doi_m \end{bmatrix} \quad (17)$$

Here $0 \leq doi_q \leq 1$ and $\sum_{q=1}^{m} doi_q = 1$

7. a_i carries out steps mentioned in algorithm 3 given in Section 4.3 and obtains DOT (degree of trust) matrix where DOT contains new degree of trust from source to its neighbors
8. a_i arranges all its neighbors in descending order on the basis of DOT matrix.
   8.1 Source keeps x most trusted neighbors from this list, where x lies in the range [min_{rec}… max_{rec}] and remove others from its neighborhood.

For the purpose of removing some agents from its neighborhood, source carries out algorithm 4. Source agent judges all its neighboring agents based on the similarity between itself and its existing neighbors and their current degree of trust by utilizing algorithm 3 (here instead of nominees, neighbor's information is utilized) and acquire matrix DOT, which is the matrix having updated degree of trust on existing neighbors. The source then sorts the list of neighbors, in descending order of their newly obtained trust stored in matrix DOT and keeps x most trusted agents in its neighborhood, where min_{rec}< k <max_{rec}. Algorithm 4 serves dual purpose of contraction of neighborhood and pruning of neighborhood on the basis of changes in source's taste of items. Without application of algorithm 3 simply removing less trustworthy agents from neighborhood do not take source and neighbor similarity into account. In this manner source can maintain number of neighbors from suggested range by applying algorithms 1 to 4.

6. Experimental Setup

Experiments were carried out to determine the appropriate range [min_{rec}… max_{rec}] of neighbors. The dataset for experiments was derived from web community of Apartmentratings.com. The data set rates thousands of apartments in USA on the seven criteria viz. Parking, Maintenance, Construction, Noise, Grounds, Safety and Office Staff. The above set of parameters describes basic features of an apartment, according to which recommender will describe the apartment and probable user will choose the apartment to live in. For experiments the data has been collected directly from the Apartmentratings Web site [13]. The dataset consists of approximately 500,000 raters who rated a total of almost 1000 different apartments at least once. The total numbers of reviews are around 1,000,000. Out of 500,000 raters, 10 different sets of 50 raters were chosen as asample to study algorithms. Thus in total 500 raters were chosen. For each set of 50 raters their corresponding 50 agents were created using JADE and profile of each user was placed in its agent. The system is implemented using Java and JADE platforms. Algorithm of recommendation generation and algorithms 1 to 4 of expansion of neighborhood and contraction of neighborhood are developed and implemented as Java classes and are integrated with the JADE platform. The interaction among different agents for developing trust relationships were implemented as agent behaviors. In the initial phase of the experiment for each of its 50 users their list of acquaintances along with the degree of trust that they can place on each other were generated randomly. According to these lists initial web of trust was spawned which is similar to web of trust in figure1 but with 50 agents. Web of trust thus contains a directed edge from an agent to all the agents in its list of acquaintances weighted by the degree of trust as reported in the randomly generated list and hence become its neighbors. This was done for each set of 50 agents.

6.1 Discussion

To discover the appropriate range of minimum and maximum number of neighbors that an agent should possess, simulations were carried out with for each set of 50 agents by making each agent as source.
In each simulation source agent initiated the process of recommendation generation with different number of neighbors in its neighborhood. For the purpose of estimation of range [min_{rec}… max_{rec}] simulations were carried out by having one neighbor and extended up to 15 neighbors in source neighborhood. For each set of 50 agents experiments were carried out and their results were documented independently as well as their average result was also recorded. Figure 2illustrates data obtained after running experiment for the first agent from first set of 50 agents.

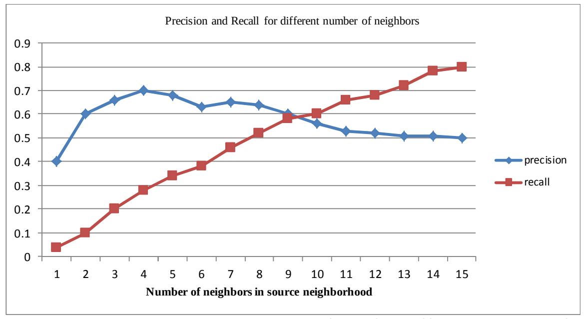

Fig. 2 Precision and recall obtained for $a_1$for different number of neighbors

The two metrics commonly used to evaluate the recommender systems are precision and recall.
Precision is defined as the fraction of the selected items that are relevant to the user's needs. It measures the selection effectiveness of the system and represents the probability that the item is relevant.

$$\text{Precision} = \frac{\text{Number of relevant recommendations retrieved}}{\text{Total number of recommendations retrieved}} \quad (18)$$

Recall is defined as the ratio of the relevant items selected to the total number of relevant items available. Recall represents the probability that a relevant item will be selected.

$$\text{Recall} = \frac{\text{Number of relevant recommendations retrieved}}{\text{Total number of relevant recommendations available}} \quad (19)$$

Considering equal error rate or equal accuracy which denotes the intersection of precision and recall curves, from the figure 2 it can be clearly seen that at the point of having 9 neighbors in a_1's neighborhood, precision and recall intersects. This graph advocates that an agent must possess 9 neighbors in its neighborhood. Similarly other runs were also carried out for first 50 agents and their readings were recorded. Another such set of 50 agents was taken to determine suitable range of neighbors and figure 3 demonstrates the result.
In the figure 3 the intersection point of precision and recall comes out to be 13. Thus according to figure 3 agents should maintain 13 neighbors in their neighborhood.

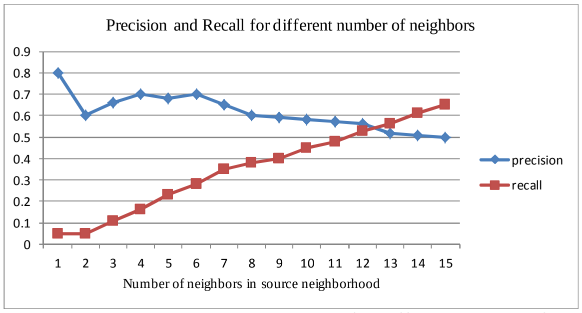

Fig. 3 Precision and recall obtained for different number of neighbors

Figure 4 demonstrates the result obtained after taking average of all the simulation of 10 sets of 50 agents in each set.

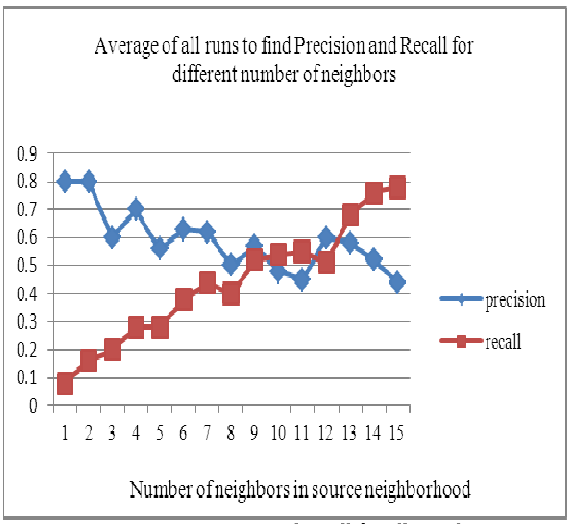

Fig. 4 Average precision and recall for all simulations

It is clearly evident from figure 4 that an agent in WoT should have between 9 to 13 neighbors which defines the optimal range for the number of neighbors that an agent should maintain. This range can be inferred from the graph as precision and recall intersect thrice: first at 9, then at 11 and again at 13. Hence, the two extreme intersections give the range on number of neighbors as **[9 …13]**.

7. Conclusion and Future Work

In this paper a suitable range of minimum and maximum number of neighbors that an agent should preserve is proposed. In order to maintain number of neighbors from this range two procedures are presented. Firstly, a process of expansion of neighborhood to assist an agent to enhance its set of recommendations by discovering good recommenders and make them its trustworthy neighbors is proposed. Secondly, a procedure of contraction of neighborhood to eliminate lesser trusted neighbors from agent's neighborhood is proposed. Proposed techniques involve discovering similarity between source agent and other agents and computation of trust on the basis of similarity factor and initial trust. Our process of expansion of neighborhood takes trust decay into account as query propagation is restricted up to one level away from source. Experimental results have demonstrated that the proposed range follows the real life pattern where one tends to have a sufficient number of close communicates, very few friends provide limited knowledge where as more than required inundate one's database.

More experiments are being conducted with some other real social network datasets to further validate the results. A feasible remedy for the situation where the source needs to enlarge its neighborhood and its direct neighbors are not able to provide adequate number of trustworthy candidates is under consideration.

References

[1] A. Jøsang, E. Gary, and M. Kinateder, "Analysing topologies of transitive trust", in 1st International Workshop on Formal Aspects in Security and Trust, 2003.

[2] A. Medic, and A. Golubovic, "P2P Appliance Calculation Method For Trust between Nodes within a P2P Network", International Journal of Computer Science Issues, Vol. 9, No. 4, No 2, 2012, pp 125-130.

[3] B. Alcalde, and S. Mauw, "An algebra for trust dilution and trust fusion", Formal Aspects in Security and Trust, 5983, Springer, 2009, pp. 4-20.

[4] C. Ziegler, "Towards Decentralized Recommender Systems", Ph.D. thesis, University of Freiburg, 2005.

[5] D. Doherty, S. Jouili, P. V. Roy, "Trust-Based Recommendation: an Empirical Analysis", in 6th SNA-KDD Workshop '12, 2012, Beijing, China.

[6] 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, No. 1, 2008, pp. 57–74.

[7] G. Guo, J. Zhang, and D. Thalmann, "A Simple but Effective Method to Incorporate Trusted Neighbors in Recommender Systems", In 20th International Conference on User Modeling, Adaptation and Personalization (UMAP), 2012, pp 114-125.

[8] G. Liu, Y. Wang, and M.A. Orgun and E.P. Lim, "A heuristic algorithm for service provider selection in complex social networks", in SCC'10, 2010, pp. 130–137.

[9] G. Liu, Y. Wang, and M.A. Orgun and E.P. Lim, Trust Transitivity in Complex Social Networks, in

Twenty-Fifth AAAI Conference on Artificial Intelligence, 2011.

[10] G. Liu, Y. Wang, and M.A. Orgun, "Optimal social trust path selection in complex social networks", in AAAI, 2010, pp. 1397–1398.

[11] H. Kaur, and D. Jain, "Modulating Friendship Trust into Context Level Trust in Trust Based Recommender Systems", International Journal of Recent Trends in Engineering, Vol 3, No. 2, May 2010, pp. 58-63.

[12] H. Mehta, SK. Bhatia, P. Bedi, and VS. Dixit, "Collaborative Personalized Web Recommender System using Entropy based Similarity Measure", International Journal of Computer Science Issues, Vol. 8, Issue 6, No 3, November 2011, pp 231-240.

[13] http://www. apartmentratings.com.

[14] J. Gimpel, K. Karnes, J. Mctague, and S. Pearson-Merkowitz, "Distance-decay in the political geography of friends-and-neighbors voting", Political Geography, Vol. 27, 2008, pp. 231–252.

[15] J. Golbeck, "Computing and Applying Trust in Web-Based Social Networks", Ph.D. Thesis, University of Maryland, 2005.

[16] J. O'Donovan, and B. Smyth, "Trust in recommender systems", In International Conference on Intelligent User Interfaces, San Diego, California, January 2005, pp. 167—174.

[17] M. Jamali, and M. Ester, "Trust Walker: a random walk model for combining trust-based and item-based recommendation", in 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 397-406.

[18] P. Bedi, H. Kaur,and S. Marwaha, " Trust based Recommender System for the Semantic Web", in 20th International Joint Conference on Artificial Intelligence (IJCAI '07), Hyderabad, India, 2007.

[19] P. Massa, and P. Avesani, "Trust-aware recommender systems", in RecSys'07, USA, 2007.

[20] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust", in WWW'04, 2004, pp. 403–412.

[21] S.MAGUREANU, "Distributed Trust-Aware Recommender Systems", Master's thesis, ICT, 2012.

[22] S. Nepal, C. Paris, and S. K. Bista, "SRec: a social behaviour based recommender for online communities", 3rd Workshop on Social Recommender System at UMAP, 2012.

[23] T. Heath, E. Motta, and M. Petre, "Person to Person Trust Factors in Word of Mouth Recommendation", in CHI2006 Workshop on Reinventing Trust, Collaboration, and Compliance in Social Systems (Reinvent06), 2006, pp. 22-27.

[24] U. Kuter, and J. Golbeck, "Sunny: A new algorithm for trust inference in social networks using probabilistic confidence model", in AAAI'07, 2007 pp. 1377–1382.

Dr. Harmeet Kaur received her Ph.D. in Computer Science from the Department of Computer Science, University of Delhi, Delhi, India in 2007. She is an Associate Professor in the Department of Computer Science, Hans Raj College, University of Delhi. She has about 15 years of teaching and research experience and has published more than 20 research papers in National/International Journals/Conferences. Her research interests include Multi-agent Systems, Intelligent Information Retrieval Systems, Trust and Personalization.

Ms. Deepali Jain is a Research Scholar and working as an Assistant Professor in the Department Of Computer Science, Hans Raj College, University of Delhi. Her research area is Knowledge Based Systems and is currently pursuing PhD under Dr. Harmeet Kaur from Department of Computer Science, University of Delhi.