School of Computing Science,
University of Newcastle upon Tyne

Trust as a key to improving Recommendation Systems

Georgios Pitsilis and Lindsay Marshall

Technical Report Series

CS-TR-875

November 2004

Trust as a key to improving Recommendation Systems

Georgios Pitsilis[1] and Lindsay Marshall

School of Computing Science, University of Newcastle Upon-Tyne
Newcastle Upon Tyne NE1 7RU, U.K.
{Georgios.Pitsilis,Lindsay.Marshall)@ncl.ac.uk

Abstract. In this paper we propose a method that can be used to avoid the problem of sparsity in recommendation systems and thus to provide improved quality recommendations. The concept is based on the idea of using trust relationships to support the prediction of user preferences. We present the method as used in a centralized environment; we discuss its efficiency and compare its performance with other existing approaches. Finally we give a brief outline of the potential application of this approach to a decentralized environment.

Keywords: *Recommendation Systems, Subjective Logic, Trust Modeling*

1 Introduction

This paper introduces a method that can be used to improve the performance of recommendation systems using trust based neighbourhood formation schemas. In recommendation systems a typical neighbourhood formation scheme uses correlation and similarity as measures of proximity. With this approach, relationships between members of the community can be found only in the case when common properties or common purpose exists, where such properties can be for example, common experiences expressed in the form of opinions about an assessed property. This requirement appears to be a problem for the formation of communities especially at the beginning of the process when in general there is a low number of individual experiences and therefore the possibility of common experiences existing is also low.

Our main idea is to exploit information, which at first glance may seem to be extraneous, in such a way that might be beneficial for the community. In a recommendation system, that benefit appears as improved accuracy, as well as improved capability in providing recommendations. We make use of any common experiences that two entities might have, to establish hypothetical trust relationships between them, through which they will then be able to relate to other entities. This would make it possible for entities which were unlinked before, to link together and use each other's experiences for their mutual benefit.

[1] Scholar of the Greek State Scholarships Foundation (IKY)

However, there are two challenges for recommendation systems which appear to be in conflict with each other : scalability and accuracy. Accuracy is proportional to the amount of data that is available but appears to work at the expense of scalability, since more time is needed to search for those data. In this paper we only deal with the first challenge, leaving the scalability issues for future work.
Our claim is that discovering trust relationships and thereby linking the users of a recommendation system together can have a positive impact on the performance of such a system.
To support our hypothesis we ran experiments on a small community of 100 nodes and compared the recommendations of our system against those that a plain collaborative filtering method would give. We also performed a comparison against the output that we would get if the choices were solely based on intuition. In our study we chose a centralized environment as our test-bed for evaluating the algorithms and for carrying out the processing of data. However, we do discuss the requirements, benefits and pitfalls if deploying it in a decentralized system.
The rest of the paper is organized as follows. In the next section there is a general description of recommendation systems, as well as related work in the field. In section 3 we explain the main idea of our work, the logic and calculus we have used in our model and we also focus on the trust modeling we have done for the purpose of our experiments. In Section 4 there are some performance measurements showing the benefits and the drawbacks of our system. In sections 5 and 6 we explain roughly how such a model might work in a decentralized environment showing some major drawbacks that have to do with the information discovery process and we also propose solutions that might help overcome these problems.

2 Motivation

2.1 Background Research

Recommender systems [1] are widely used nowadays and in simple terms their purpose is to suggest items, usually products, to those who might be interested in them. The main idea is to get the users that participate in such system correlated based on the opinions they have expressed in the past and then to provide as suggestion lists of products that might be of interest, or in the simplest form, predictions of ratings of services or products they want to know about.
Recommender systems often exist as services embedded into web sites which provide support for e-commerce activities. *epinions.com* [2], *amazon.com* [3] and *ebay* [4] are some of the most popular commercial sites. Some others, such as *Grouplens* [5] have been built with the sole purpose of supporting research activities in this area.
Technologies that have been applied to recommender systems include *Nearest-neighbor* (which includes *Collaborative filtering* (CF)), *Bayesian networks*, and *Clustering*. Bayesian networks create a decision tree based on a set of user ratings.

Despite their speed in providing recommendations they are not practical for environments in which user preferences are updated regularly. In clustering, users are grouped by their similarity in preferences and predictions are made regarding the participation of a user in some cluster. In the case of participation in multiple clusters the prediction is a weighted average. As is shown in [6], algorithms based on Clustering have worse accuracy than nearest-neighbor therefore pre-clustering is recommended.

The basic idea behind CF is to make predictions of scores based on the heuristic that people who agreed (or disagreed) in the past will probably agree (disagree) again. Even though such a heuristic can be sufficient to correlate numerous users with each other, systems that have employed this method still appear to be highly sparse and thus are ineffective at making accurate predictions all the time. This ineffectiveness is proportional to how sparse the dataset is. By Sparsity we mean a lack of data required for a CF system to work, where in this specific case the data are in the form of experiences which users share with each other through the system. Sparsity appears mostly because users are not willing to invest much time and effort in rating items.

Conventional recommendation systems face other problems such as the *cold start problem* [7] and their *vulnerability to attacks.* The latter comes from the centralized nature of a collaborative filtering system and the fact that there are always users that have malicious intent and want to influence the system. The attacker can simply create a fake user with very similar preferences to that of the targeted user and thus he/she becomes highly influential to the victim. The cold start problem, is due to the low number of ratings that new users contribute to the system who thus becoming isolated and cannot receive good quality recommendations. Developing other types of relations between the users, especially new ones, could help increase their connectivity base and thus their contribution to the system.

2.2 Trust and Reputation in Computing Systems

Trust and *Reputation* have always been a concern for computer scientists and much work has been done to formalize it in computing environments [8]. In computing, *Trust* has been the subject of investigation in distributed applications in order to enable service providers and consumers to know how much reliance to place on each other. *Reputation* is a commonly held belief about an agent's trustworthiness. It is mainly derived from the recommendations of various agents.

Yahalom et. al in [9] distinguish between directly trusting an entity about a particular subject and trusting an entity to express the trustworthiness of a third entity with respect to a subject. These two types of trust are known as direct and indirect (or derived) trust. This raises the issue of how one can traverse a whole chain of intermediate entities to find a trust value for a distant one. In fact, there is a debate as to whether it is valid or not to consider transitive trust relationships. Even though it has been shown that trust is not necessarily transitive [10], there are various requirements such as the context, otherwise known as the trust purpose, that need to be specified and which indicate the ability to recommend [11]. This ability, if it exists, makes indirect trust possible. Assuming that this ability is present in a long chain then a recommendation can be made, as indirect trust can be calculated along the chain.

2.3 Trust Modeling

Trust can be thought as the level of belief established between two entities in relation to a certain context. In *uncertain probabilities* theory [12] *belief* is expressed with a metric that is called *opinion.* Because opinions are based on observations, there is always imperfect knowledge and therefore it is impossible to know for certain the real (objective) behavior of the examined entity. This theory introduces the notion of uncertainty to describe this gap of knowledge or else the absence of belief and disbelief. Uncertainty is important in trust modeling, as it is always present in human beliefs and thus is suitable for expressing these kinds of beliefs.
A framework for artificial reasoning that makes use of the uncertainty in the expression of beliefs is called *Subjective Logic* [13]. It has its basis in uncertain probabilities theory and provides some logical operators for combining beliefs and deriving conclusions in cases where there is insufficient evidence.
From a probabilistic point of view there would be both a certain amount of belief and disbelief which is used to express the level of trustworthiness with absolute certainty. As that absolute certainty can never exist, the uncertainty property (u) has been introduced to fill this gap and deal with the absence of both belief and disbelief. The probabilistic approach would treat trustworthiness by observing the pattern of an entity's behavior and using only two properties belief (b) and disbelief (d) where b+d=1, $b,d \in [0,1]$. Binary calculus assumes statements of trust as dual-valued; either *true* or *false*. As such, Subjective Logic can be seen as an extension of both binary calculus and probability calculus. The relationship between b,d and u is expressed as b+d+u=1 which is known as the *Belief Function Additivity Theorem* [13].
Subjective Logic also provides the traditional logical operators for combining opinions (e.g. $\wedge,\vee$) as well as some non-standard ones such as *Suggestion* and *Consensus* which are useful for combining series of opinions serially or in parallel.
A complete reference about the algebra of *Subjective logic* and on how the algebra is applied to *b*,*d* and *u* can be found in [14].
Even though opinions in the form (b,d,u) are more manageable due to the flexible calculus that opinion space provides, evidence is usually available in other forms, that are easier for humans to understand. In [13] there is a mapping between Evidence Spaces to Opinion Spaces based on the idea of coding the observations as elements of the Beta Distribution probability function. In this approach the uncertainty property (u) appears to be exclusively dependent on the quantity of observations. [15] has an alternative mapping that uses both quantitative and qualitative measures to transform observations into opinions.
In contrast, other similarity based approaches [16] use the idea of linking users indirectly with each other using predictability measures, but these have not been tested on real environments.

As we mentioned, the requirement for trust to become transitive in long chains requires that a common purpose will exist along the chain. Only the last relationship should concern trust about a certain purpose and all the other trust relationships in the chain should be with respect to the entities' recommending abilities for the given purpose.
It is worth mentioning the existence of another approach to making recommendation systems trust-enabled [17] which does not distinguish between functional and recommended trust.

3 Our Approach

3.1 Using Trust in Recommendation Systems

As we mentioned in 2.1, in standard collaborative filtering, the correlation of ratings is done on a nearest-neighbour basis, which means only users who have common experiences can be correlated. In that schema only knowledge within a radius of one hop from the referenced node can become useful. For example, in the simple scenario of figure 1, where 3 services are experienced by 4 entities, using the standard method, there is no way for any knowledge from entity A to be used for providing recommendations to entity D.

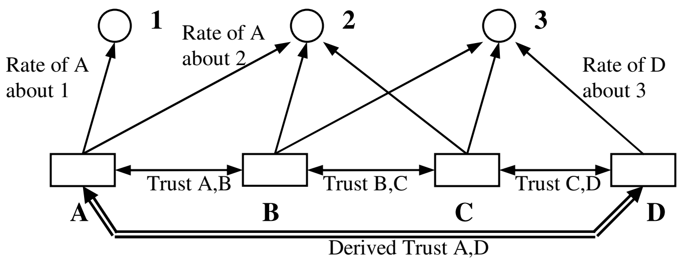


Figure 1.Using Trust to link A B C and D together

Our idea is to exploit information from any experiences that can be reached beyond the barriers of the local neighborhood for the benefit of the querying entities. We deal with this issue by utilizing the trust that could exist between the entities and in this way build a web of trust within the system.
Then, those entities that may have experienced the services in question would be reachable through the web of trust, thus providing additional information to the querying entities for making predictions about services they might like, but where no relevant experiences have been found for them within their local neighbourhood.
However, this requires some way of mapping the experiences of each entity into trust measures and in the opposite direction, transforming the derived trust into some form of information that can provide useful recommendations.

Once all these issues are shorted our, the quality of the recommendation system should improve since more queries can now be answered and thus avoid the problems of Sparsity.

3.2 Our Trust modeling

In general, trust models are used to enable the parties involved in a trust relationship to know how much reliance to place on each other and there are a few models that have been proposed to calculate trust, for instance [13][18].

The problem that emerges when Trust is to be used in a recommendation system is the fact that entities involved provide ratings of items rather than their trust estimates of other entities. That means, making the model trust enabled would require that all this information that has been expressed in the form of ratings be transformed into trust values, and this requires a transformation method.

In [15] we proposed a technique for modeling trust relationships between entities derived from evidence that characterize their past experiences. Our model aims to provide a method for estimating how much trust two entities should place on each other, given the similarities between them. In this model, the entities are considered as more similar if they can more accurately predict the other's ratings. Predictions can be done using Resnick's [19] formula or some alternative to this [16][6].

$$p_{a,i} = \bar{r}_a + \sum_{u=1}^{n} w_{a,u}(r_{u,i} - \bar{r}_u) \qquad (1)$$

where $\bar{r}_a$ is the average rating of the querying user, $w_{a,u}$ is the *Coefficient* of the similarity correlation of user α with user u (which appears as weight in the deviation), $\bar{r}_u$ is the average rating of each of the n entities that provide recommendations, $r_{u,i}$ is their ratings of each of the n and $p_{a,i}$ is the predicted rating.

As can be seen from the formula, the prediction is dependent on the number of correlated entities n and becomes noisy and unreliable when 5 or less entities are involved. Hence, higher accuracies become possible with the incorporation of more entities in the calculation.

In contrast to modeling trust using the Beta distribution function [13], our similarity based modeling technique can also be used in the opposite way for estimating similarities given the trust between the entities. Using this characteristic, a querying entity that can receive ratings about some distant entities for which it can make a trust estimate through the graph, will be able to use them in its prediction schema.

4 Our Experiments

As we mentioned in section 2.3, there is a requirement for a common trust purpose that has to be met in order to regard trust as transitive. In the trust graph idea we have used to bring users together, we assumed as the common purpose the involved party's ability to recommend. This ability comes from the way the trust relationships have been formed. Hereafter, in this experiment we assume that the entities, in fact, do have this ability to provide recommendations as soon as they appear to have a common taste on things.

We performed a series of tests to examine how efficient our system might be if applied to a real recommender system. Efficiency is measured as how successfully the system can predict the consumer's preferences. In our experiments we used data from the publicly available dataset of the MovieLens project [20]. MovieLens is a film recommendation system based on collaborative filtering, established at the University of Minnesota. The dataset that is publicly available contains around one million ratings of movies made by 6,040 users who joined the MovieLens in the year 2000. In our experiment we used a subset of 100 users which comprises only around 13,000 ratings.

To avoid poor performance due to the noisy behavior of the Pearson Correlation, we applied some filtering to the existing relationships. Therefore, those relationships which were built upon 5 or less common experiences were not considered in our calculations.

The dataset also contains timestamps for every rating indicating when the rating took place, but that information was not considered at all in our correlations since at this stage we intended to study the static behaviour of the model. The timestamps might be useful in some future experiment if used as a secondary criterion for choosing ratings to be considered in the trust relationships. For example, only ratings that have been issued by both counterparts within a certain amount of time will be considered in a trust relationship.

In our analysis, we demonstrate how such system would perform in comparison to standard collaborative filtering. We also applied a comparison schema against a system that involves no use of recommendation system, but where the users make the choices themselves by using their intuition. For this comparison, every user's predictions were guided alone by personal, past experiences. Needless to say such a schema has meaning only to those users that share some significant number of personal experiences.

We introduce two notions that will be used in our measurements:

Computability: We define this as the total number of services for which a user can find opinions through the trust graph, divided by the total number of services that have been rated by all counterparts in the sample. This normalization value should be seen as a performance limit, as no more services can be reached by any of the counterparts. The Computability value is specific to a particular user since the number of services that can be reached is absolutely dependent on users' position in the graph as well as the number of their own experiences.

Recommendation Error (E): We define this as the average error of users when trying to predict their own impressions of those services they can reach when using the reputation system. It can be defined as the prediction rating divided by the rating that is given after the experience. Similar to *Computability*, this measure is also specific to a particular user.

To provide a unique metric of effectiveness, we also introduce the *Normalized Coverage* factor F. This measure combines *Recommendation Error* and *Computability* into a single value and is expressed as:

$$F = (1 - \overline{E}) \cdot C \qquad (2)$$

where:

$\overline{E}$ is the average recommendation error for a particular user and C is the Computability value for that user. F represents how much a user benefits from his participation in the community. High values of F should mean that participation is beneficial for the user.

4.1 Testing Method

For evaluation purposes we used two algorithms, one for the calculation of *Computability* and another one for the *Recommendation Error*. Due to the static nature of the data that we used in the experiment there was no way to simulate a real environment of users experiencing services. For that reason, to be able to measure the difference between a prediction and the actual experience, we used a technique called *leave one out* as our metric. In this method one rating is kept hidden and the system tries to predict it.
The pseudocode we used to evaluate the Recommendation Error of the recommendations is given in figure 2.

```
Let S the set of all services
Let f the filter used in the trust propagation
Let hop the number of hops the trust can be propagated
Let U the set of all users in the group
For each user $U_A$ in U {
  Let $S_A \subset S$ the experiences of $U_A$
  For each service s in $S_A$ {
    Let r = R[$U_A$,s]                /* The rate given from $U_A$ on s */
    Let $B_S \subset U$ the users that have also experienced s
     For each user $U_B$ in $B_S$ {
        Trust$_{A \to B}$= f(path $_{A \to B}$, f , distance$_{A \to B}$ < hop)
        CC $_{A,B}$ = f(Trust$_{A \to B}$)      /* similarity between $U_A$ & $U_B$ */
     }
     Let Sp = f(CC$_{A,i}$), $\forall\, i \in Bs$   /* The predicted rating about s */
     Let Err = f(r,Sp)
   }
   RecomError = Average(Err)
}
```

Figure 2. Pseudocode for evaluating the Recommendation error

The difference between the real rating – mentioned here as post-experience – and the predicted rating gives the error. Setting k=1 in the algorithm returns the prediction error for the plain CF method which is based on examining the nearest neighbours only. In the same way, k=0 can give the error if users were doing the choices guided by their intuition alone. In our experiments we run tests for k ranging from 0 to 3.

For *Computability* (or *Coverage)*, we also ran evaluations for values of k=0,1,2,3. The pseudocode of the algorithm we used is shown in figure 3.

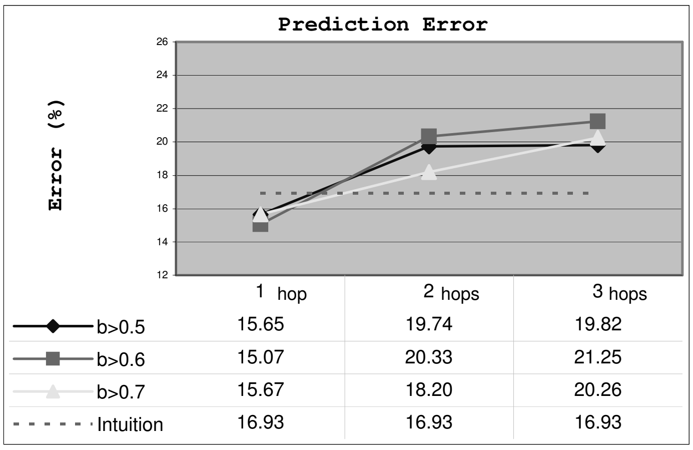
```
Let S the set of all services
Let f the filter used in the trust propagation
Let hop the number of hops the trust can be propagated
Let U the set of all users in the group
For each user $U_A$ in U {
 Let $S_A \subset S$ the set of services that $U_A$ has experienced
 For each user $U_B$ in U {
   $Trust_{A \to B}$ = f($path_{A \to B}$, f, $distance_{A \to B}$ < hop)
   If $Trust_{A \to B}$ > 0 {                /* B is reachable by A  */
     Let $S_B \subset S$ the set of services that $U_B$ has experienced
     $S_A = S_A + S_B$         /* Add $S_B$ to A's potential experiences */
    }
  }
  Coverage = $S_A$ / S
}
```

Figure 3. Pseudocode for calculating the coverage

For the calculation of trust between any two entities in the trust graph we used a parser to discover the accessible trust paths between the trustor entity and the trustee. Then we applied subjective logic rules (the *consensus* and *recommendation* operators) to simplify the resulting graphs into a single opinion that the trustor would hold for the trustee at distance k. It was necessary to do this separately for every individual entity to prevent any opinion dependency problems [14] that can be caused when hidden topologies exist. Therefore, the calculation of the resulting trust was left to be carried out by every trustor individually.
Moreover, in cases where trust paths couldn't be analyzed and simplified further by just using these two operators, we applied a simple pruning technique to remove those opinions that were found to cause problems in the simplification process. In this study, the selection of the pruned links was done at random, but we leave for future work an extensive study of the consequences of using pruning in trust calculations as well as the formulation of a policy that would minimize these consequences.

4.2 Results

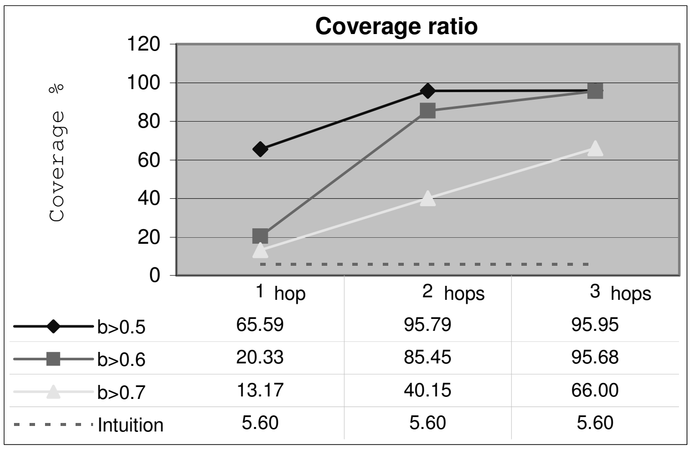
Figures 4,5 and 6 show the results from the experiments we performed. Figure 4 shows how Prediction Error changes with regard to hop distance and various belief filtering policies in the propagation of trust. In total, we compared three filtering policies (*b*>0.5 , *b*>0.6 and *b*>0.7), where *b* is the belief property. In this filtering policy, trust is not allowed to propagate to entities that are not considered trustworthy as defined by the filter. So, path exploration proceeds up to the point where it is found that the belief property of some neighboring entity does not exceed the value set on the filter.
The same diagram also shows the results from the plain CF method (1 hop) as well as the case where users make choices using only their intuition. For the latter, there is no categorization for various trust filters since there is no use of trust at all. The results represent average values taken over the series of 100 entities.
It seems that, on average, the intuitive rating appears to have the lowest error, but as we will see, this criterion is inadequate to be used for judging. An equally interesting fact is that in our method (Hop distance>1) the error is not affected significantly as the hop distance increases, which means there is no loss of precision if using the trust graph.



	1 hop	2 hops	3 hops
◆ b>0.5	15.65	19.74	19.82
■ b>0.6	15.07	20.33	21.25
▲ b>0.7	15.67	18.20	20.26
- - - - - Intuition	16.93	16.93	16.93

Figure 4. Accuracy of Recommendations.

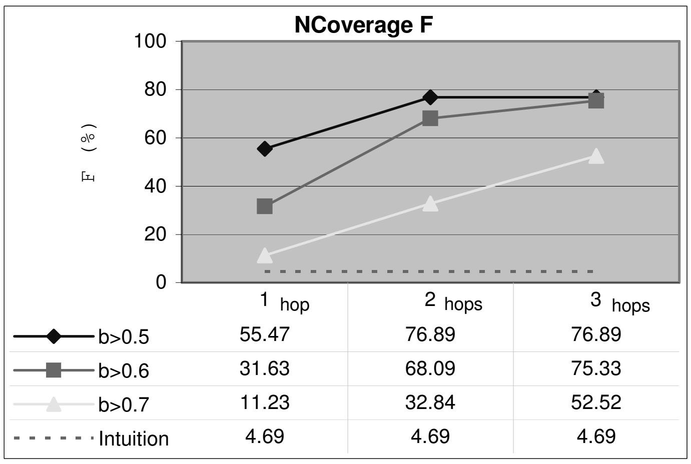
Figure 5 shows the average Coverage ratio, that is, the number of reachable services for the group of 100 users divided by the total number of services that opinions can be expressed about. In all cases, our method appears to perform better against both the intuitive choice and the plain CF method. A strong filtering policy though has a negative impact especially for short hop distances, whereas applying an average filter (0.6) seems to improve the situation. (hop=2)



	1 hop	2 hops	3 hops
◆ b>0.5	65.59	95.79	95.95
■ b>0.6	20.33	85.45	95.68
▲ b>0.7	13.17	40.15	66.00
- - - Intuition	5.60	5.60	5.60

Figure 5. Computability of Recommendations

Finally, figure 6 presents the *Normalized Coverage Factor* we introduced and which can be thought as the total gain from using some policy. From the graph it seems that the participants do not benefit when strong filtering policies are applied. Strong filtering though is less consuming in resources due to the simpler graphs that have to be explored and be simplified. Therefore, such comparison without including the cost compared to the benefit would not be fair. We leave as future research a full performance analysis that would find the best policy.



	1 hop	2 hops	3 hops
◆ b>0.5	55.47	76.89	76.89
■ b>0.6	31.63	68.09	75.33
▲ b>0.7	11.23	32.84	52.52
- - - Intuition	4.69	4.69	4.69

Figure 6. Normalized Coverage F

5 Discussion

The increase in computability that our method can achieve, also has a positive effect on the reduction of Sparsity. Our measurements show a significant fall of sparseness of 9.5%. The original 100 user dataset was found to be 97.85%. sparse. This calculation of sparseness is based on the 100 users we used in the evaluations over the whole set of items (6040), where the total number of ratings expressed by those 100 users was 12976.

By using our method, only 30% of the users benefited by using the trust graph. The remaining 70% were those who received the same benefit as when using the plain CF system (1 hop). This is because in the dataset it was likely for two users to have common experiences and this is dependent on how clustered the user communities in the dataset are. For example, if there were more than one clustered communities of users, then on average the benefit of using the graph would be higher because a very small number of users would be enough to bridge the gap between those separate clusters and thus to increase the number of recommendations that can be received on the other side.

That extra 30% benefit characterizes the potential of the proposed system with respect to the dataset used in the experiment. Speaking in terms of sparseness, that potential over the 100-user base, constituted a dataset that was 88.33% sparse.

The results justify the explanation we gave in section 2 saying that the plain method suffers from reduced coverage due to the small number of ratings that close neighbors can provide. This is because the nearest-neighbors algorithms rely upon exact matches.

For the prediction error, a comparison against a random choice of ratings instead of predicting them shows our method to be better even for hop distances of 3. Using our dataset, random generated values would give error rates as low as 24.5%, but such a comparison would be unfair for two reasons, first because it requires access to global knowledge which is unlikely to be possible, and second, because the error is highly dependent on the distribution of ratings over the classes of rates.

As can be seen, even though our method increases the system output by increasing the quality of recommendations as compared to the plain method, the algorithms do not scale to large amounts of data and thus performance degrades for a large number of users. In other words, the design will not lead to a system capable of providing quality recommendations in real-time. Even if the complex and expensive computation of direct trust vectors is done off-line there will be a bottleneck in calculating the indirect trust relationships and discovery of trust paths. This is because a direct trust vector needs to be recalculated whenever a new rating is introduced by either of the two sides in a trust relationship. However, these re-calculations could be done off-line as background processes, preserving the computing power for the graph analysis. This is feasible since in such recommendation systems the user and the item data do not change frequently.

For the above reasons, the method does not seem suitable for use in centralized systems. Cacheing techniques might provide some extra speed in calculations, provided that changes in the virtual trust infrastructure will not happen frequently.

6 Future Issues

In the future, we intend to perform a comparison of our method with other alternatives such as Horting [16] or others based on Dimensionality Reduction such as Singular value Decomposition [21. As regards the depth we chose to do the graph analysis, we anticipate performing more analyses using greater depths than the 3 hops we used in this experiment. This would help us to study how the performance increases with the depth of search and also find the optimum depth given the high computational load that depth searching requires.

As we mentioned in the Discussion section, our method is not suitable for application in centralized systems because it is highly compute intensive. However, a promising solution to overcome this weakness would be to restructure the centralized system as a peer-to-peer recommendation system. In such an approach the benefit is two fold. It provides distributed computational load as well as higher robustness and lowers vulnerabilities to the kinds of attacks we described in section 2. This architecture is also closer to the natural way that recommendations within groups of people take place.

As regards the requirement for a common purpose to exist in order for trust relationships to be used in a transitive way, we intend to alter the assumptions we have made about the existence of common purpose and re-run the experiment using pure recommender trust in the transitive chains. That requires though, that we can somehow model the trust placed on a recommender's abilities for the given purpose, using the existing evidence.

We also plan to investigate the model from a graph theoretical perspective and examine how the Clustering Coefficient of the trust graph might affect the quality of recommendations. Also a close analysis of every user separately could show better which users benefit most from their contribution in the system.

7 Conclusion

We proposed a method that is based on the idea of using trust relationships to extend the knowledge basis of the members of groups so they can receive recommendations of better quality. In this study we applied a model that uses quantitative and qualitative parameters to build trust relationships between entities based on their common choices. We used algebra for relating users together through the transitive trust relationships that can be built between them and we extended in this way their neighboring basis. For the evaluations we used real data from a publicly available centralized recommendation system and we presented some preliminary results about the performance of our method, we discussed the benefits and the pitfalls. Our first results show that despite the fact that the method seems incapable of providing recommendations in real time, it seemed to improve the efficiency of the system, which translates into increased computability without significant impact on the accuracy of predictions. We also pointed out how the disadvantages could be overcome if the method is applied in decentralized environments such as peer-to-peer.

8 References

[1] P.Resnick – H.R.Varian, “Recommender Systems”, Communications of the ACM, 40(3): 56-58, 1997

[2] http://www.epinions.com

[3] http://www.amazon.com

[4] http://www.ebay.com

[5] P.Resnick - N.Iacovou - M.Suchak - P.Bergstrom - J.Riedl, "Grouplens. An Open Architecture for Collaborative filtering of Netnews, from Proceedings of ACM 1994, Conf. On Computer Supported Cooperative Work.

[6] Breese, J.S. Heckerman, D. Kadie,C. (1998). “Emperical Analysis of Predictive Algorithms for Collaborative Filtering”. In Proc. of the 14th Conference on Uncertainty in Artificial Intelligence pp 43-52.

[7] D. Maltz and K.Ehrlish. “Pointing the Way: Active Collabortive filtering”. In Proc. Of CHI-95,

[8] S.Marsh, “Formalizing Trust as Computational concept”, PhD Thesis, University of Stirling, Scotland 1994.

[9] R.Yahalom, B.Klein, T.Beth, ”Trust relationships in secure systems – A Distributed authentication perspective”, In Proc. of the 1993 IEEE Symposium on Research in Security and Privacy, p 152. pages 202-209,Denver,Colorado 1995.

[10] Cristianson B.,Harbison W., “ Why isn’t Trust Transitive?”, In Proc. of the Security Protocols Workshop, p171-p176, 1996

[11] A.Jøsang – E.Gray – M.Kinateder, “Analyzing topologies of Transitive Trust”, In proceedings of the Workshop of Formal Aspects of Security and Trust, (FAST 2003), Piza September 2003.

[12] G.Shafer, "A Mathematical Theory of Evidence", Princeton University Press. 1976

[13] A.Jøsang, “A Logic for Uncertain probabilities”, International Journal of Uncertainty, fuzziness and Knowledge based systems, Vol.9,No.3, June 2001.

[14] A.Jøsang, “An Algebra for Assessing Trust in Certification Chains”, In proceedings of NDSS’99, Network and Distributed Systems Security Symposioum, The Internet Society, San Diego 1999.

[15] G.Pitsilis. – L.Marshall., “A model of Trust Derivation from Evidence for User in recommendation Systems”, School of Computing Science, University of Newcastle Technical Report Series, November 2004.

[16] Aggarwal, C.C. Wolf, J.L., Wu K. and Yu, P.S. (1999). “Horting Hatches an Egg: A New Graph-theoretic Approach to Collaborative Filtering”. In Proceedings of the ACM KDD’99 Conference. San Diego, CA, pp.201-212

[17] P.Massa – P.Avesani, ”Trust-aware Collaborative Filtering for recommender Systems”, CoopIS/DOA/ODBASE (1) 2004: 492-508

[18] Rahman.A – Heiles.S, “Supporting trust in Virtual Communities”, Proceedings of International conference on System Sciences, Jan 4-7 2000, Hawaii

[19] Resnick,P. and Varian, H.R. (1997). “Recommender Systems”. Special issue of Communications of the ACM. 40(3).

[20] Bradley N. Miller, Istvan Albert, Shyong K. Lam, Joseph A. Konstan, John Riedl. (2003). “MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System”. InProceedings of ACM 2003 International Conference on Intelligent User Interfaces (IUI'03) (Accepted Poster), January 2003.

[21] Sarwar.B.M.,Karypis.G.,Konstan.J.A.,Riedl.J.T, “Application of Dimensionality Reduction in Recommender System-A Case Study”, WebKDD Workshop, August 20, 2000.