International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 3, March 2016

A Distributed Collaborative Filtering Framework for DHT-Based P2P Network

[1] Partha Sarathi Chakraborty [2] Dr. Sunil Karforma
[1] Lecturer, University Institute of Technology [2] Reader, Dept. of Computer Science,
[1][2] The University of Burdwan
[1] psc755@gmail.com [2] sunilkarforma@yahoo.com

Abstract: **Collaborative filtering is the most common technique for designing e-commerce recommender systems. Traditional recommender systems based on collaborative filtering works basically in a centralized way. So they are not scalable for large networks. In this paper we have designed a distributed collaborative filtering framework for a structured P2P network where user profiles are distributed over the nodes of the network. At the same time, the computation for generating recommendation is also distributed over the nodes. A distributed clustering layer has also been proposed to the framework to reduce the communication overhead and at the same to make the system more scalable.**

I. INTRODUCTION

Collaborative filtering] [Resinck et. al. 1994],[Herlocker et. al. 2000] is the most common approach for designing e-commerce recommender systems. It works by building a database of items with users' opinions on them. Then a specific user is matched against this database in order to find her neighbors, those with whom she shares similar tastes. Over the time demand for designing distributed recommender systems are increasing due to several reasons. First, as online users are growing over time, in case of centralized solution, huge computational resources are needed to be installed by the organization for generating good quality recommendation. This can be avoided by distributing the storage of rating data as well as computation of the predicted rating among the nodes of the underlying network. Second, one of the major challenges in designing recommender system is to handle the sparsity problem. [Weng 2009] provides a B2B solution for reducing this problem by suggesting sharing of rating information among various recommenders operating in the same domain. The information enrichment will lead to produce recommendation of high quality. Third, as have been mentioned in [Lam et. Al. 2006], customers may not feel comfortable to share the information regarding their personal choices with a central authority. They may prefer to keep their profile in their local computer due to privacy reason. As distributed recommender system does not have any central authority it provides good solution for the privacy issue. From another point of view, the customer may doubt about the recommendation generated by the organization. So, going one step further, customer may wish to choose the algorithm also for generating the recommendations.

II. RELATD WORK

[Miller et. al. 2004] introduced an item-based collaborative filtering algorithm (they name it PocketLens) for five different distributed environments. The central server architecture stores ratings in a central server and computation for model building and recommendation generation is done in customer's node. Random discovery architecture and Transitive Discovery architecture uses the protocol of Gnutella (www.gnutella.com) peer-to-peer architecture for neighborhood selection. Latter has an improvement over the previous one by learning of the neighborhood incrementally as a result of entering new users into the system. Their last reference architecture is based on structured peer-to-peer network. Two Chord based schemes are proposed. In the first scheme, each user's pseudonym and rating is stored in the network as a (key, value) pair. The second scheme stores item identifier and the corresponding rows of the item-item matrix in the nodes. In both the schemes, a model is built gradually in the customer's node and computation for recommendation generation is made locally-not it has been distributed.

[Weng 2009] advocates for cooperation of multiple business organizations of similar nature and proposed a distributed recommender system made of multiple recommenders from different organizations. A user-based random walk approach has been adopted by [Kermarrec 2010]. In a number of works [Oka 2004], [Wang 2006], [Tveit 2001] distributed recommender systems have been designed for P2P networks. Han et. al. [Han 2004] distribute the rating data among the nodes of a DHT based overlay network using buckets. Each bucket is identified by the <item_id, rating_value> pair and contains user details of those users who has rated the item with item_id by that score (rating_value). They propose a heuristic algorithm for data

integration from the network. They also propose significance refinement (SR) and unanimous amplification (UA) for improving accuracy and scalability of the system. [Sorge 2007] proposed a chord based implementation of item-based collaborative filtering algorithm where security and privacy issues have been considered.

III. CHORD ARCHITECTURE

Chord [Stocal et. al. 2001] is a distributed lookup up protocol for peer-to-peer networks. It provides a mechanism to identify a node which stores a particular key. Chord does not store the data itself. To know the location of the data, the key for that data value is passed to the lookup service which identifies the node storing the key and the associated data. Chord is fully distributed in the sense that every node has equal importance and provides the same lookup service. The lookup service is based on consistent hashing [Karger 1997], [Lewin 1998]. Key and node identifiers are ordered in a identifier circle modulo 2^m where m is the length of the identifiers. A key is assigned to a node whose value is same as the key value or it is assigned to the next node in the identifier circle. Figure x. shows an example chord ring having 7-bit key identifiers and node identifiers. The key K5 and K20 is assigned to node N24 and key K60 is assigned to N90.

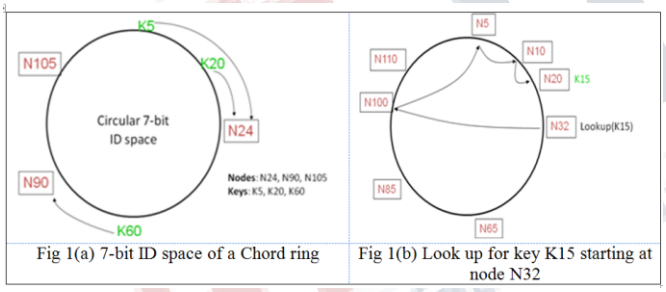

Fig 1(a) 7-bit ID space of a Chord ring

Fig 1(b) Look up for key K15 starting at node N32

To limit the number of nodes to be searched for finding the location of a key, every chord node maintains a finger table with at most m entries. The ith record in the finger table (called ith finger) of node n contains three fields called finger[i].start, interval and the successor of that finger. The field finger[i].start refers node $n+2^{i-1}$ modulo 2^m. interval field designates the interval [finger[i].start, finger[i+1].start). figure xx shows the lookup process originated at node N32 for key K15 stored in node N20. The use of finger table in the lookup operation increases scalability and each key lookup requires at most O(*log(N)*) messages.

IV. PROPOSED FRAMEWORK

A. Basic Framework

The general framework for distributed recommender system based on chord protocol has been shown in figure 2.

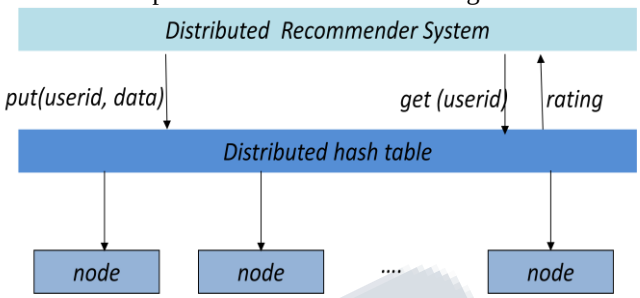

Fig.2 Distributed Collaborative Filtering Framework

The user profiles are distributed among the peers of the Chord network. The node where a user profile is stored is determined by hashing the user id of that user. In addition to the user profiles, a data structure called item table is maintained for each individual item in the system. Location of the item tables is determined by hashing the item id of the corresponding item. Virtually two Chord rings are maintained. One is a ring of identifiers for user-ids and other is the ring of identifiers for the item-ids. Identifiers of both the rings are mapped separately to the same set of physical nodes. So, each node in the distributed system stores a set of user profiles and a set of item tables. An item table stores user-id of the users who purchased the corresponding item.

A. Algorithm for inserting a new rating, $r_{i,j}$

Steps:

1. Using hashing, generate the identifiers for the corresponding user U_i and the item I_j .
2. Following chord protocol identify the node where the user profile is stored.
3. Insert the new rating in the user profile if the profile exists. Otherwise create the new profile.
4. Using the identifier for item I_j locate the node where the item table for I_j is stored.
5. Append user id of user U_i to the item table of I_j .

B. Algorithm for Recommendation generation

Originating node, that is the node of active user starts the process. Suppose U_a needs recommendation for I_p .

Steps:

For each item, I_i rated by the active user, U_a -

1. Calculate hash value of I_i and send messege $< U_a, I_p, I_i >$ to appropriate node.
2. Item table of that node will give the list of users who rated I_i.
3. For each user U_m in the list, send message $< U_a, I_p, I_i, U_m >$ to node having profile of Um.

   (U_a is given in the message so that the node can send result message directly to originating node.)
4. Nodes return message $< U_m, R_{m,p}, R_{m,i}, mean(U_m), sd(U_m) >$ provided U_m has rated I_p.

   ($R_{m,p}$ will be used in prediction calculation, $R_{m,i}, mean(U_m), sd(U_m)$ will be used in similarity calculation)

   (U_m is used in gathering information of all items for a single user)
5. Originating node can calculate similarity locally as follows-

$$sim(U_a, U_m) = \sum_{commonitemsi} (Ra,i - mean(Ua))(Rm,i - mean(Um)) \Big/ sd(Ua) * sd(Um)$$

6. N Nearest neighbors, U_{nni} are chosen.
7. Predicted rating is calculated locally as-

$$Ra,p = mean(Ua) + \sum (Rm,p - mean(Um) * sim(Ua,Um) \Big/ \sum sim(Ua,Um)$$

V. PERFORMANCE OF THE SYSTEM

In case of centralized systems, Mean absolute Error is the most common metric for measuring the performance of the recommender systems. As the same collaborative filtering algorithm is used, the proposed system will exhibit the same accuracy as the centralized system. The improvement comes from mainly two angles-distributing the storage of the user profiles and distributing the computation of recommendation generation.

A. Communication Cost

Suppose there are n users and m items. If, on an average, each user has rated αm items, then αm messages will be sent to nodes for consulting the item tables. Again, suppose, on an average, βn users have rated each of the items, αm. So βn messages will be sent to nodes having profile of those βn users. As a result βn result messages will be sent back to originating node. Total communication cost is αm + αm* 2βn or αm(1+2βn).

VI. EXTENSION TO THE BASIC FRAMEWORK

A. Reducing Communication Overhead and increasing Scalability; Distributed Clustering Layer:

The basic framework involves a large number of message exchange among the nodes in the generating the recommendation. To reduce this communication overhead further, a distributed clustering layer can be added to the framework. For example, a P2P version of the K-Means algorithm [Bandyopadhyay et. al. 2006] can be used. The distributed clustering algorithm groups the users into different clusters and every node maintains the clustering information. The clustering is done offline. Whenever an user request for recommendation, the originating node builds a complete list of other users who belongs to the cluster of the active user and sends it as part of the request message to other nodes. This not only reduces the local computations in the first-level nodes but also limits the number of second-level nodes to be communicated by the first level nodes.

VII. CONCLUSION

In this article, we propose a framework for distributed collaborative filtering algorithm for a distributed hash table (DHT) based P2P network where each peer stores a fraction of the whole rating database in the form of a set of user profiles. Other than that peers also maintain a data structure named Itemtable which stores list of users rated for a particular item in order to calculate the similarity among users and predicted rating in a parallel and distributed way.

REFERENCES

1. [Bandyopadhyay et. al. 2006] Sanghamitra Bandyopadhyay, Chris Giannella, Ujjwal Maulik, Hillol Kargupta, Kun Liu, Souptik Datta: Clustering distributed data streams in peer-to-peer environments. Inf. Sci. 176(14): 1952-1985 (2006)

2. [Herlocker et. al. 2000] J. Herlocker, J. A. Konstan, J. Riedl, "Explaining Collaborative Filtering Recommendations", in Proceedings of ACM Conference on Computer Supported Cooperative Work, Philadelphia, PA, 2000.

3. [Karger 1997] Karger, D., Lehman, E., Leighton, F., Levine, M., Lewin, D., AND Panigrahy, R. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing* (El Paso, TX, May 1997), pp. 654–663.

4. [Kermarrec 2010] A.-M. Kermarrec, V. Leroy, A. Moin, and C. Thraves. Application of random walks to decentralized recommender systems. Principles of Distributed Systems, 2010.

5. [Lam et. Al. 2006] Shyong Lam, Dan Frankowski, and John Riedl. Do you trust your recommendations? an exploration of security and privacy issues in recommender systems. In Gnter Mller, editor, Emerging Trends in Information and Communication Security, volume 3995 of Lecture Notes in Computer Science, pages 14–29. Springer Berlin / Heidelberg, 2006.

6. [Lewin 1998] Lewin, D. Consistent hashing and random trees: Algorithms for caching in distributed networks. Master's thesis,

7. [Han 2004] P. Han, B. Xie, F. Yang, and R. Shen, "A scalable P2P recommender system based on distributed collaborative f iltering," Expert Systems WithApplications, vol. 27, no. 2, pp.203-210 2003.

8. Department of EECS, MIT, 1998. Available at the MIT Library, http://thesis.mit.edu/.

9. [Miller et. al. 2004] Bradley N. Miller, Joseph A. Konstan, John Riedl: PocketLens: Toward a Personal Recommender System. ACM Transactions on Information Systems 22 (July 2004).

10. [Oka 2004] T. Oka, H. Morikawa, and T. Aoyama, "Vineyard: A collaborativefiltering service platform in distributed environment," in SAINT-W '04: Proceedings of the 2004 Symposium on Applications and the Internet- Workshops (SAINT 2004 Workshops). Washington, DC, USA: IEEEComputer Society, 2004, p. 575.

11. [Resinck et. al. 1994] Resinck., P., Neophytos, I., Mitesh, S., Peter, B., John, R., 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 ACM conference on Computer Supported Cooperative Work, Chapel Hill, North Carolina, United States, p.175-186.

12. [Sorge 2007] Sorge. C., A Chord-based Recommender Sys tem, Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on.

13. [Stocal et. al. 2001] Stocal, I., et al. (2001). Chord: a scalable peer-to-peer lookup service for Internet applications. In: ACM SIGCOMM, San Diego, CA, USA, pp. 149–160.

14. [Tveit 2001] A. Tveit, "Peer-to-Peer Based Recommendations for Mobile Commerce", in Proceedings of the International Workshop on Mobile Commerce, Rome, Italy, 2001

15. [Wang 2006] J. Wang, J. Pouwelse, R. Lagendijk, and M. R. J. Reinders, "Distributed collaborative filtering for peer-to-peer file sharing systems," in Proceedings of the 21st Annual ACM Symposium on Applied Computing (SAC06), 2006.

16. [Weng 2009]

17. Weng, Soloman, Xu, Yue, Li, Yuefeng, & Nayak, Richi (2009) Towards Information Enrichment through Recommendation Sharing. In Cao, L (Ed.) *Data Mining and Multi-agent Integration*. Springer, United States of America, pp. 103-126.