Collaborative Filtering via Epidemic Aggregation in Distributed Virtual Environments

Patrick Gratz
University of Luxembourg
6, rue Coudenhove-Kalergi
L-1359 Luxembourg
patrick.gratz@uni.lu

Jean Botev
University of Luxembourg
6, rue Coudenhove-Kalergi
L-1359 Luxembourg
jean.botev@uni.lu

Abstract

The ever-increasing amount of available information in today's digital society necessitates inline techniques for determining the most relevant content. Collaborative filtering (CF) systems have proven to be an adequate means for reducing informational overload and generating useful recommendations. Current systems are predominantly built on centralized or, more recently, structured Peer-to-Peer (P2P) approaches. However, in order to apply collaborative filtering to large-scale distributed virtual environments (DVEs) in unstructured networks with substatially higher user numbers, different approaches are necessary.
Within this paper we present a collaborative filtering algorithm for DVEs utilizing epidemic data aggregation based exclusively on local information. Designed to be extremely scalable, it creates recommendations in a transparent way by distributing an accumulated view of favorite ratings to interacting users. The algorithm is intended for deployment in the HyperVerse - a self-organizing middleware service for large-scale DVEs - for generating and managing rating predictions of object favorites. Evaluation results show that, in terms of quality, locally aggregated predictions converge well on those obtained from an idealized global view.

1 Introduction

As the information load of today's society increases substantially every day, obtaining the most relevant data becomes more and more difficult. By generating recommendations for items based on the taste of like-minded users, collaborative filtering is a well-established technique to reduce information overload [17]. Prevalent systems using collaborative filtering mostly depend on huge centralized databases to store user preferences or item-item similarity matrices. In particular large online retailers like e.g. Amazon gather personal information or characteristic patterns over time either explicitly by the user giving a rating on an item or by implicitly counting clicks, viewing time or other characteristic data to generate preferences. Some of these systems combine CF with content-based techniques, which often suffer from limited item categorizations when employed alone.
Instead of utilizing centralized databases, recommendations can also be obtained through finding the necessary data in a transparent, distributed fashion. Such an approach becomes crucial when dealing with extremely large-scale, decentralized scenarios as for instance next-generation DVEs. This paper proposes a novel collaborative filtering scheme based on the epidemic aggregation of rating data for the use in such environments which scales in terms of user and item numbers. It creates recommendations for object or location favorites in a lightweight fashion by distributing an accumulated view of favorite ratings between interacting users. The mechanism is meant to be integrated directly into the HyperVerse infrastructure [3], a P2P-based self-organizing middleware service for massively distributed virtual environments, and utilizes its internal management update messages. This, along with capped list sizes, assures that only an insignificant amount of extra network load is induced for the CF. Additionally, the presented algorithm is not based simply on a distributed item-item matrix or preference profiles: ratings and recommendations are created and managed locally according to the social bias of the user.

The remainder of this paper is structured as follows: Section 2 features related work in the context of collaborative filtering, P2P overlay networks and distributed environments. Section 3 gives a short introduction to the HyperVerse, which incorporates the filtering algorithm. Section 4 then covers the CF algorithm itself, while Section 5 describes the experiments and evaluations performed with our filtering approach on different datasets in comparison to an

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

idealized complete knowledge scenario. Finally, a summary and conclusion of the proposed approach and possible future work are given in Section 6.

2 Related Work

In this section we introduce and discuss existing research in the context of collaborative filtering in distributed environments.

[20] considers a method for creating a scalable recommendation system for mobile commerce using P2P systems. The main idea of the proposed approach is to transform the problem of extracting recommendations through collaborative filtering into a general search problem in scalable P2P systems like Freenet or Gnutella. Thereby, a query (a vector with votes on products) is broadcasted from the querying node to all its neighbor peers. Once a peer receives a query it calculates the proximity to other cached queries. If the proximity is higher than a certain threshold, the cached voting vector is sent back; otherwise the query is disseminated further. For sparse voting vectors, the authors propose a binary interpolative compression algorithm. Furthermore, to improve the performance and quality of recommendations, they introduce an approach for the clustering of similar peers.

Kim et al. propose a similar approach in [15]. In addition to the own user profile, each peer manages a friend list which stores peers with similar interests. In order to initially find such friends for a peer, the system utilizes a flooding mechanism. A receiving peer calculates cosine similarity between the received and its personal user preferences. If the calculated value is higher than a certain threshold, an identifier of the receiving peer together with the own ratings are sent back. Otherwise, the receiving peer simply forwards the request. After this broadcast, the returned top n similar peers are kept in the friends list of the source peer. For each peer in such a friend list, an overlay network ("friend network") can be created. In order to periodically update a peers' friends list, the authors propose to initiate the flooding mechanism over the friend network instead of flooding the entire network again.

In [10] an algorithm called pipeCF is presented, which manages user databases and calculates rating predictions in a decentralized way within a P2P overlay network based on distributed hash tables (DHT). In order to reduce the number of returned nearest neighbors for an active user (who should be included in the prediction algorithm for the local user), the authors additionally introduce a techniw called significance refinement (SR) which improves prediction accuracy.

In all the above mentioned and other similar approaches [16, 9] user profiles are collected either by using a broadcasting mechanism or by using DHTs to map unique keys to profiles. This requires additional network traffic overhead, resulting in severe scalability issues when extended to the envisioned global scale [21].

A different approach for distributed collaborative filtering in P2P file sharing systems is also presented in [21]. Unlike the above approaches, this system implicitly "learns" user interests from users' interaction data. Consequently, the list of downloaded items is considered as the user profile. As a key concept, each item stores a so called "buddy table" containing a list of other items relevant to itself. These buddy tables represent a self-organizing semantic overlay that implicitly clusters similar items. Furthermore, updates of the buddy tables only occur when items are downloaded. Thus no additional network traffic is generated. Due to the fact that this approach presumes the exchange of multimedia files in order to create and update the corresponding buddy tables - a common aspect for file sharing systems - it is not applicable for our scenario, either.

A collaborative filtering technique in a mobile tourist information system for visitors of a music festival, based on spatio-temporal proximity in social contexts, is proposed in [6]. The presented idea utilizes a user-based CF technique and calculates similar users via a location- and time-based proximity measure, i.e. two users are considered to be similar if they consume the same event simultaneously. Rating information between these similar users is exchanged via an ad-hoc P2P interaction. Furthermore, rating information is only exchanged if a peer resides in another peer's coverage area for a certain period of time. The defined similarity measure has one definite drawback though: users consuming the same periodic event at different times still share interests, but are not considered to be similar. In future work, the authors intend to investigate how their CF approach can be extended in order to exchange ratings between users in spatial but not temporal proximity. Additionally, they plan to evaluate the introduced CF system at the Edinburgh Fringe festival.

In [22] Yang et. al introduce a decentralized collaborative filtering for P2P systems based on a distributed probabilistic item ranking model. To overcome the problem of a missing centralized database of user preferences in their P2P system, the authors developed an epidemic preference exchange protocol called BuddyCast. No Evaluation has been performed yet, but is intended as future work.

In [18] a further approach inter alia based on an epidemic spreading of user preferences is presented and also evaluated. In their study the authors show how to reach comparable prediction accuracies in a mobile scenario as well as in a complete knowledge scenario. An evaluation is presented, however there are two major shortcomings. Firstly, limited resources of mobile devices have not been considered as unlimited cache sizes are presumed. Secondly, no realistic mobility model has been used, but an

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

idealized data exchange pattern with disjoint pairs per iteration.

3 The HyperVerse

Before introducing the CF algorithm in detail, this section will give a brief overview of the HyperVerse project. Investigating fundamental principles suitable for the realization of extremely large-scale and highly interactive DVEs, the project aims at creating a self-organizing and sustainable middleware service as a basis for future virtual worlds like e.g. a 3D Internet.
One key feature of the HyperVerse middleware is the underlying two-tier architecture shown schematically in Figure 1: A loosely-coupled, statistical P2P client overlay on top of a federated and highly structured public server infrastructure which explicitly includes client resources at the network's edges. This concept differs from existing hosting approaches for MMVEs (Massively Multiuser Virtual Environments) or MMOGs (Massive Multiplayer Online Games), which are still mostly based on a centralized server infrastructure and therefore suffer from severe scalability issues when extended to a larger or even global scale. Data dissemination in the HyperVerse is based on a Torrent-like technology [5], taking into account virtual geography and thus exploiting access locality. Clients are interconnected based on their proximity in virtual space via a geographic indexing service, effectuating specific qualities of the network, e.g. the power-law property, and permitting self-adaptation of the statistical structure. For more details on the infrastructure and the underlying technologies, we refer to [3].

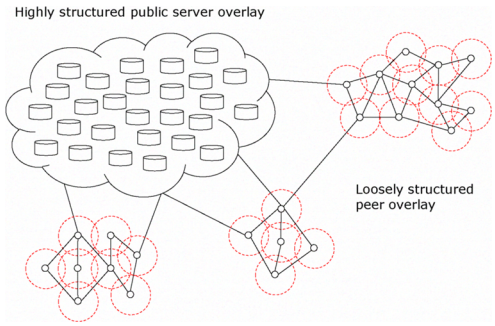

Figure 1. Two-tier HyperVerse architecture.

An important technical aspect of the HyperVerse infrastructure to mention for our CF approach however is the reciprocal exchange of management messages during peer interactions. Here we augment the existing logic by adding data necessary for the collaborative filtering algorithm in a piggy-back fashion as described in detail in the following section. The efficient provision and management of the basic background services is not focus of this paper but subject to ongoing other research [8].

4 CF via Epidemic Aggregation

In order to calculate a rating prediction for a certain item, centralized user-based CF methods use their global knowledge to determine a set of top N similar users to the target user that have rated this item. Subsequently, only the profiles of these N users are used to estimate the rating of the target user for this specific item.
However, the P2P infrastructure of the HyperVerse as well as user and oject dynamics in virtual environments do not allow for centralized database-backed global knowledge in the manner of item or preference matrices, since there is no way of efficiently storing or accessing them. Thus, in our CF approach we calculate predictions directly within the peers based on local information only by weighted averaging of rating deviations as described in [11]. Due to the fact that the quality of the calculated predictions depends on a critical set of locally available similar user preferences, we need an efficient mechanism to determine such user preferences for each peer. The following subsection introduces an information exchange protocol to efficiently aggregate the most similar users for each peer.

4.1 Algorithm Description

Our aggregation algorithm is based on an epidemic protocol and exchanges user information in a gossiping manner. Especially in the context of large-scale distributed systems such protocols emerged as an efficient communication paradigm maintaining scalability and simplicity [14]. In order to keep additional network load low, we designed our algorithm as a piggy-back mechanism to already existing network messages periodically induced by so-called interactions. These could be either explicitly triggered by user-to-user interactions like e.g. a chat or physical actions within one's virtual area of interest (AoI), or implicitly triggered through interest group memberships, status messages of buddies and so on. Along with a moderate cap for the additional data, a maintainable and limited overhead for the CF is assured. Furthermore, under the assumption that a user within the HyperVerse will mainly interact with other like-minded users in the virtual environment, this mechanism leads to a native ratio of exploitation and exploration via a more frequent exchange between like-minded users and a rather sporadic exchange with other, unknown users. The pseudo code in Listings 1 and 2 showcases the key elements and general functionality of our algorihtm. It operates on a model generating interaction pairs of peers based on a snapshot of the social relations between them. The

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

model is described in detail in section 5.
At first, each peer maintains a local cache containing its own profile information, a list of most similar user preferences `Buddies` and frequently received less or not similar profiles `Others` as depicted in Figure 2. This structure differs from existing approaches which do not distinguish between profiles relevant to a peer itself and other data for altruistic distribution to help increase effectiveness. In order to limit the cache size corresponding to the available memory resources within a peer, sizes of both lists are capped. During an interaction peers exchange their local profile information containing `Buddies`, `Others` and their own preferences. Afterwards, each peer updates its local information via the received data.

```
FUNCTION main()
  FOR each interaction pair x,y
    data_x = x.getCFData()
    data_y = y.getCFData()
    x.updateState(data_y)
    y.updateState(data_x)
  END FOR
END FUNCTION

FUNCTION updateState(CFdata)
  FOR each profile in CFdata
    sim = calculateSimilarity(profile)
    IF sim >= similarityThreshold
      IF profile is in others THEN
        others.Remove(profile);
      END IF
      result = updateBuddies(profile, sim)
      IF result is NOT true THEN
        updateOthers(profile)
      END IF
    ELSE
      IF profile is in buddies THEN
        buddies.Remove(profile);
      END IF
      updateOthers(profile)
    END IF
  END FOR
END FUNCTION
```

Listing 1. Core Algorithm (Pseudo Code)

Before incorporating a received profile into the local cache, the similarity between this profile and the actual user profile is calculated. For similarity calculation we used the Pearson correlation coefficient, which is a very accurate performing measure in existing CF approaches [4]. If the resulting similarity value is higher than or equal to a specified threshold, the corresponding profile will be inserted into `Buddies`. Otherwise, the profile will be inserted into `Others`. Once the `Buddies` or the `Others` list reaches its limit, further profiles are inserted by applying the following replacement strategies.

4.2 Replacement Strategies

Once the `Buddies` list fills up, a new profile will replace an existing one if and only if its similarity value is higher than the least similar profile. If the similarity is not high enough, the profile will get a chance to be added into the `Others` list. Replacement here is more complex, with two different levels of importance based on a counting mechanism. The list is sorted top down according to this value, resulting in the most frequently exchanged user profiles building the head of the list.

```
FUNCTION updateBuddies(profile, sim)
  result = false
  IF buddies is full THEN
    lsbv = buddies.getLeastSimilarBuddyValue()
    IF lsbv < sim THEN
      buddies.removeLeastSimilarBuddy()
      buddies.add(profile, sim)
      result = true
    END IF
  ELSE
    buddies.add(profile, sim)
    result = true
  END IF
  RETURN result
END FUNCTION

FUNCTION updateOthers(profile)
  IF profile is in others THEN
    others.incrementCounter(id)
  ELSE
    IF others is full THEN
      randomValue = dice(0,1)
      IF randomValue > 0.5 THEN
        others.removeRandomCandidateFromTail()
        others.add(profile, 1)
      END IF
    ELSE
      others.add(profile, 1)
    END IF
  END IF
END FUNCTION
```

Listing 2. Cache Update (Pseudo Code)

The tail - configurable in its size - constitutes the pool of replacable profiles with lower counter values. This quantitative distinction is made in order to be able to distribute mostly profiles of so-called *super nodes* from the list. The profiles on the lower level, i.e. the tail of the list, are then replaced using a random strategy. This, along with the altruistic character of the `Others` list, generates a certain "background emission" of broader-range preference data for improving chances of receiving relevant profiles.

5 Evaluation

In order to evaluate our algorithm, we performed experiments in extensive simulation runs with existing rating databases utilizing exchange models based on social network graphs. On the one hand, we picked the MovieLens[1] data set, using a scale-free model for graph generation. On the other hand, we chose the Epinions[2] data set with substantially higher user numbers and more sparse data. Graph

[1] http://www.grouplens.org
[2] http://www.trustlet.org/wiki/Extended_Epinions_dataset

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

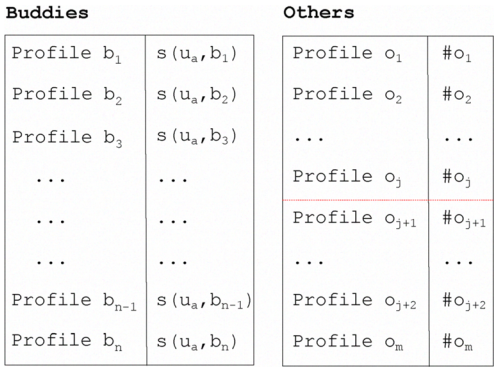

Figure 2. Local cache of user u_a. Buddies are ordered corresponding to their similarity while the order on the right list is based on a counting mechanism with only profiles at the tail of this list being replaceable directly.

generation here is based on explicit trust/distrust information already included in the set. The following sub-sections describe in detail both the data sets and exchange model, as well as the evaluations of the actual algorithm based on this data.

5.1 Data

The MovieLens data set consists of 10.000.000 ratings for 10.681 movies by 71.567 users. All ratings have been given on a scale from 1 to 5 with each user rating at least 20 movies. The Epinions data set consists of 13.668.319 ratings for 1.560.144 articles by 132.000 users. It further includes 841.372 trust statements (717.667 trusts and 123.705 distrusts). For both sets, the data has been split into a training and a test set with the training set containing 80 percent and the test set the remaining 20 percent of the data.

5.2 Data Exchange

Due to the fact that the performance of gossip-based protocols is highly sensitive to the overlay network topology itself [13], a realistic topology is crucial for evaluating our algorithm. Simple uniform and randomly organized topologies would not be suitable, as they do not sufficiently reflect the structure and properties of existing social networks, such as a power-law distribution of node degrees, low average distance and moderate clustering coefficient [7].
For mapping the social relationships between users in the MovieLens set, we chose the Barabási-Albert model [1] to create scale-free [2] power-law graphs. According to this model we then applied the two following simple rules to generate the social network graph itself:

Growth: Starting with a small number of nodes m_0, at every discrete time step add a new node with $m(\leq m_0)$ edges that link the new node to m different nodes already present in the graph.

Preferential attachment: The probability Π that a new node will connect to a node i depends on the degree k_i of node i, so that

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}. \quad (1)$$

Each node in the resulting graph represents a single user within the virtual environment. Using Barabási-Albert here ensures that the generated number of links corresponds to real social coherences, with only a small amount of users constituting super nodes with a huge list of acquaintances, while the majority only has a few.
For the experiments with the Epinions data set, we used a trust-based graph model generated utilizing the trust and distrust statements already contained in the set. Again, each node in the graph represents a single user within the virtual environment. In addition, each edge in this graph has a binary weight, where a positive weight represents trust and a negative one distrust. While the generated scale-free graph only ensures that the generated number of links corresponds to real social coherences, the trust-based graph additionally maps common interests between users.

These graphs each represent a snapshot of social relations between peers at a given time, with interactions being processed in a second step. To generate the interaction pairs, we select one node as primary candidate with probability for selection proportional to its degree. This way, the power-law property is extended also to the interactions, in favor of super nodes. To further assure a correct representation of user behaviour, subsequently with a specified probability p a previously unknown, not yet connected node is chosen as second participant to the interaction. Vice versa, already connected neighboring nodes are chosen with probability $1-p$, with the only exception that negative-weighted links are not being considered on the trust-based graph.

5.3 Evaluation Criteria

To measure the quality of our algorithm, we have chosen the following two basic criteria in our experiments:

Firstly, for predictive accuracy we used the mean absolute error (MAE) metric , which is defined as the absolute average difference between predicted and actual ratings:

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

$$MAE = \frac{\sum_{i=0}^{M} |pred(i) - r(i)|}{M}, \qquad (2)$$

with M being the total number of items in the testset having a calculated prediction, while $pred(i)$ and $r(i)$ are the predicted as well as actual ratings for an item i .
Secondly, besides a good predictive accuracy, the size of the item-domain for which a recommender system can give a prediction has a crucial impact on its suitability as underlined by Herlocker et al. in [12]. For this purpose, prediction coverage was selected as further evaluation criterion.

5.4 Experiments and Results

For investigating the behavior of our approach, we compared an idealized global view, i.e. a complete knowledge scenario where the peers have access to all data, to our epidemic algorithm based on local and limited caches with different parameter distributions.
As mentioned already in Section 4, capping the message size is crucial for scalability within the targeted scenario. For this reason, we restricted the additional overhead induced by our CF algorithm on a management message to a maximum of 40 kilobytes. With each entry coded into a 64 bit value (for instance 2 bits representing the rating itself, and the remaining bits for object and user identifiers), this would allow for at most 5.000 of them in a single message. For our experiments we furthermore limited the number of entries per profile to 40 (twice the amount of the minimum ratings given by any user), and then accordingly chose the following `Buddies/Others` ratios: 120/0, 100/20 and 80/40. For simplicity, all local entries are being exchanged during an interaction, i.e. the local cache size equals the maximum message overhead.
To assess the quality and use of our cache structure with the altruistic `Others` list, we investigated the aforementioned list size ratios utilizing a set of different similarity thresholds. A higher threshold implies better correlates within the `Buddies` list which is expected to improve prediction quality. On the downside, limiting the `Buddies` through a very high threshold is likely to lead to a substantial decrease of prediction coverage.

We simulated 1 million (MovieLens) and 2 million

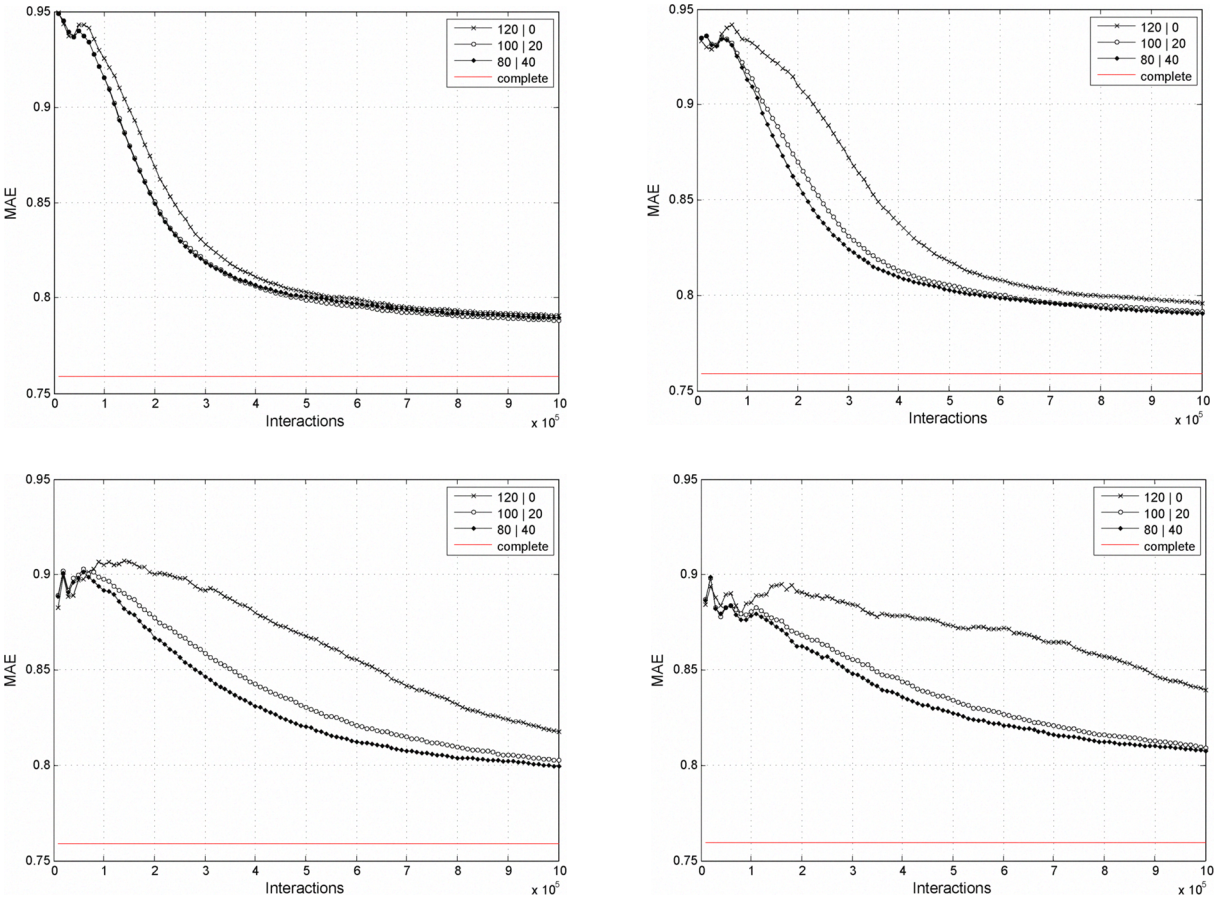

Figure 3. Accuracy development under different thresholds (0.0 - 0.3), MovieLens.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

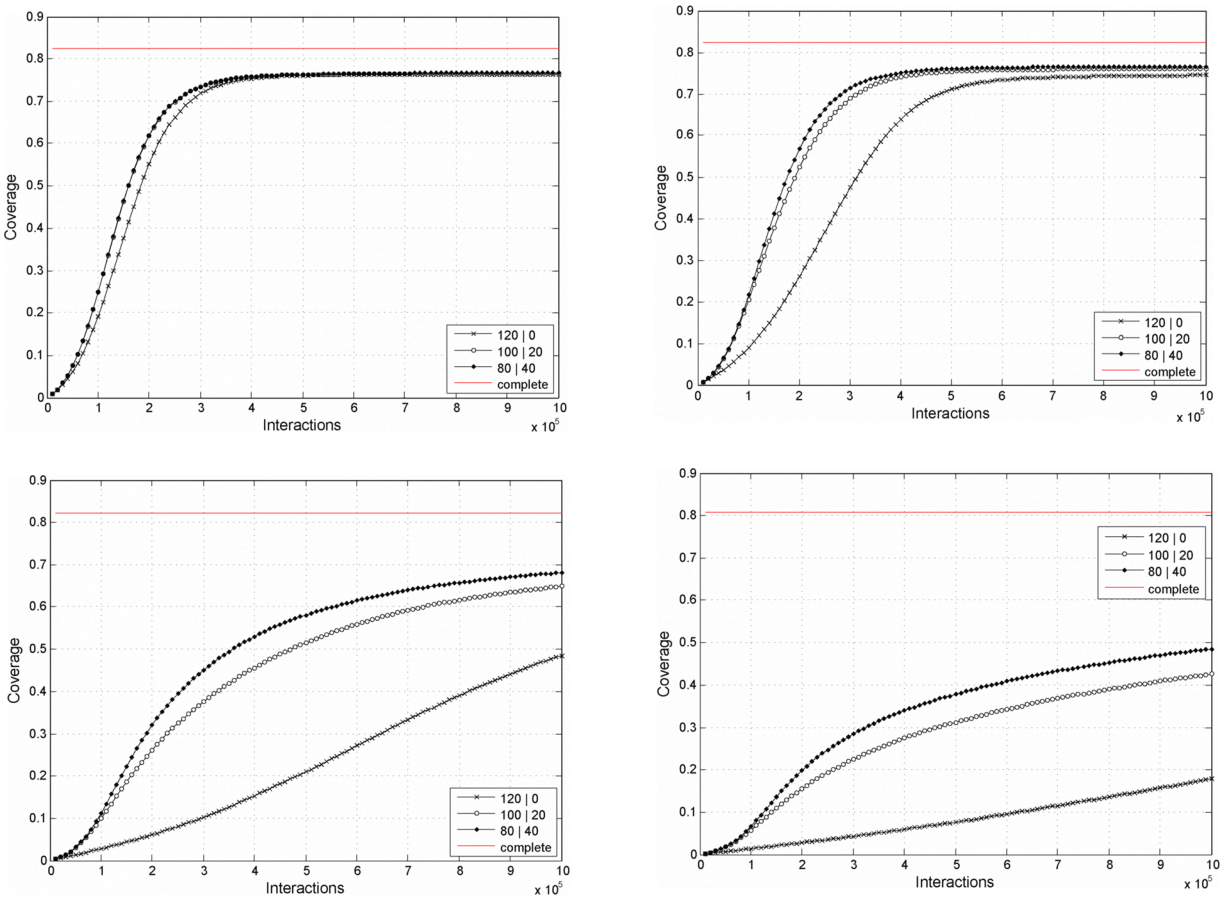

Figure 4. Coverage development under different thresholds (0.0 - 0.3), MovieLens.

(Epinions) interactions respectively, calculating predictions every 10.000 steps based on the 80 most similar profiles. Following that, we then measured accuracy and coverage for these predictions as stated above.

Figure 3 and 4 showcase the development of the prediction accuracy and coverage for the different ratios in comparison to a complete knowledge scenario (red line) with equal size restrictions and parameters for the MovieLens dataset. The red line marks the value that would be reached if all users had the profile information of their - globally seen - 80 most similar users. Similarity thresholds for the `Buddies` list starting from 0.0 to 0.3 are covered in both figures from top left to bottom right. We did not simulate with all possible threshold values, as the coverage would decrease to an unacceptable level.

As shown in Figure 3, prediction accuracy converges relatively fast on the optimum within our simulation interval (on average, every user has 14 interactions). The complete knowledge value is not yet reached within this interval. The expected positive effect of the rather altruistic `Others` list reveals itself more and more with ascending similarity threshold. However, over time the achieved accuracy is slightly worse than with lower thresholds. This is likely due to "random noise" generated by the fact that correlates in the data set are too close in terms of their predictive quality. Albeit for a significantly smaller subset of MovieLens, this effect has also been already reported in [11].

The coverage converges best with a similarity threshold of 0.0 as depicted in Figure 4. With the threshold ascending, the algorithm also clearly benefits from the usage of the `Others` list which improves convergence behavior substantially compared to a non-altruistic strategy, even if its size constitutes less than 17 percent of the maximum capacity as in the 100/20 ratio. In addition to correlates in the data set being rather close in terms of predictive quality, there is a high number of low correlates which ultimately lead to the growing distance to the complete knowledge scenario when excluded.

Overall, on the MovieLens data set the CF algorithm shows a well-convergent behavior, which is also sustained by the test runs we made on the set without limiting the rating number as depicted in Figure 5. Both accuracy and coverage values converge extremely fast and at first even exhibit slight superiority to the complete knowledge scenario

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

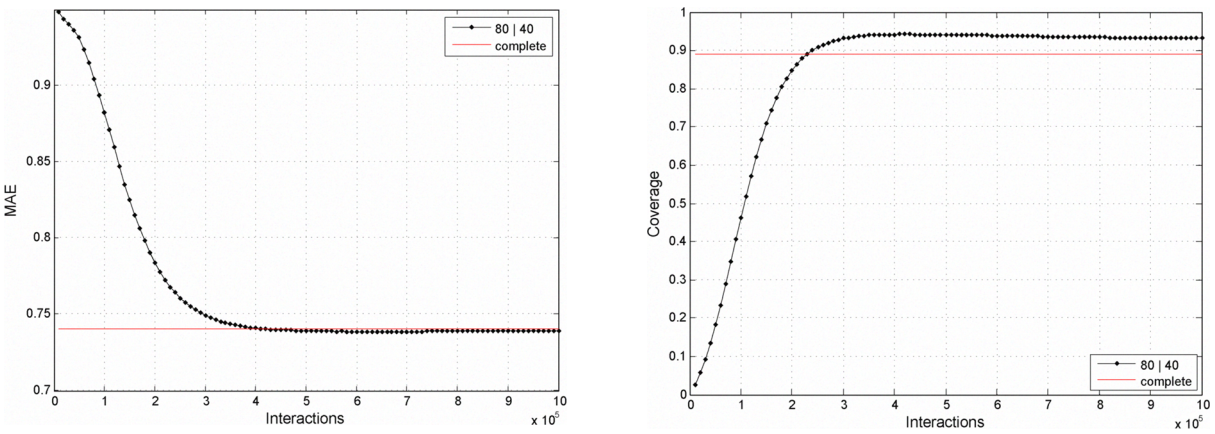

Figure 5. Accuracy and coverage development in unrestricted case, MovieLens.

but then settle on the same value. Considering the MAE, the initial burst is related to the random noise phenomenon mentioned before. On the coverage side, the most similar peers do not have to necessarily be the heaviest raters and coverage thus increases with less similar buddies before settling on the final value.

We have conducted the same experiments on the Epinions dataset, particularly because it is one of the largest available sets and allowed for the promising opportunity of trust-based modeling. Because of the majority being so-called "cold start users" with extremely few ratings, after two million simulated interactions the coverage reaches a microscopical 0.07 percent, which does not allow for reasonable measurements in view of our targeted scenario. Also the MAE value measured for prediction accuracy is questionable, because of the unbalanced distribution of ratings which focus almost exclusively on the two highest possible values (10.421.534 rating 5, 2.063.927 rating 4).

6 Conclusion and Future Work

The presented CF algorithm provides a lightweight and transparent way to generate recommendations within DVEs using an epidemic aggregation mechanism. Designed to be embedded in the underlying infrastructure of the HyperVerse, only a small and most of all limited amount of further network load is induced since the CF is realized as a piggy-back mechanism of already existing messages. Our distributed algorithm operates solely based on locally available information.

Evaluation results show that, in terms of quality, the aggregated predictions converge well on those obtained from an idealized global view. For achieving this, it is sufficient to reserve only a small fraction of the available bandwidth to exchange and store some altruistic profile data.

It is intended to carry out further evaluations for examining different aspects of the approach in more detail. For instance, the effects and selectional strategies needed in case of the local cache size superseding the maximum message overhead within an interaction is an interesting field to explore.

Also, it is planned to perform additional experiments within the simulation environment TopGen [19], utilizing different models to generate other interaction patterns, along with deploying and testing the recommendation system - regarding user numbers obviously on a smaller scale - also live.

7 Acknowledgment

Funding for this research was provided in part by the Fonds national de la Recherche (Luxembourg). We would also like to thank GroupLens (MovieLens) and TrustLet (Epinions) for the provision of the original data sets.

References

[1] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–101, 2002.

[2] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[3] J. Botev, M. Esch, A. Höhfeld, H. Schloss, and I. Scholtes. The hyperverse - concepts for a federated and torrent-based "3d web". In *Proceedings of the 1st International Workshop on Massively Multiuser Virtual Environments at IEEE Virtual Reality (MMVE 2008)*, 2008.

[4] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI 98)*, pages 43–52. Morgan Kaufmann, 1998.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278

[5] B. Cohen. Incentives build robustness in bittorrent. In *Proceedings of the 1st Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[6] A. de Spindler, R. D. Spindler, M. C. Norrie, M. Grossniklaus, and B. Signer. Spatio-temporal proximity as a basis for collaborative filtering in mobile environments. In *Proceedings of the CAISE*06 Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS '06)*, 2006.

[7] A. H. Dekker. Realistic social networks for simulation using network rewiring. In *Proceedings of the International Congress on Modelling and Simulation (MODSIM 2007)*, 2007.

[8] M. Esch, J. Botev, H. Schloss, and I. Scholtes. Gp3 - a distributed grid-based spatial index infrastructure for massive multiuser virtual environments. In *Proceedings of the International Conference on Parallel and Distributed Systems (ICPADS 2008)*, pages 811–816. IEEE Computer Society, Los Alamitos, CA, USA, 2008.

[9] P. Han, B. Xie, F. Yang, and R. Shen. A scalable p2p recommender system based on distributed collaborative filtering. *Expert Systems with Applications*, 27(2):203–210, 2004.

[10] P. Han, B. Xie, F. Yang, J. Wang, and R. Shen. *A Novel Distributed Collaborative Filtering Algorithm and Its Implementation on P2P Overlay Network*, volume 3056/2004 of *Lecture Notes in Computer Science*, pages 106–115. Springer, 2004.

[11] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR '99)*, pages 230–237. ACM, New York, NY, USA, 1999.

[12] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[13] M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems*, 23(3):219–252, 2005.

[14] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 482–491, 2003.

[15] B. M. Kim, Q. Li, and A. E. Howe. A decentralized cf approach based on cooperative agents. In *Proceedings of the 15th international World Wide Web conference (WWW '06)*, pages 973–974. ACM, New York, NY, USA, 2006.

[16] B. N. Miller, J. A. Konstan, and J. Riedl. Pocketlens: Toward a personal recommender system. *ACM Transactions on Information Systems*, 22(3):437–476, 2004.

[17] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international World Wide Web Conference*, pages 285–295. ACM, New York, NY, USA, 2001.

[18] R. Schifanella, A. Panisson, C. Gena, and G. Ruffo. Mobhinter: epidemic collaborative filtering and self-organization in mobile ad-hoc networks. In *Proceedings of the 2008 ACM conference on Recommender systems (RecSys '08)*, pages 27–34. ACM, New York, NY, USA, 2008.

[19] I. Scholtes, J. Botev, M. Esch, A. Höhfeld, H. Schloss, and B. Zech. Topgen - internet router-level topology generation based on technology constraints. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2008)*, pages 1–10. ACM, New York, NY, USA, 2008.

[20] A. Tveit. Peer-to-peer based recommendations for mobile commerce. In *Proceedings of the 1st international workshop on Mobile commerce (WMC '01)*, pages 26–29. ACM, New York, NY, USA, 2001.

[21] J. Wang, J. A. Pouwelse, R. L. Lagendijk, and M. J. T. Reinders. Distributed collaborative filtering for peer-to-peer file sharing systems. In *Proceedings of the 2006 ACM symposium on Applied computing (SAC '06)*, pages 1026–1030. ACM, New York, NY, USA, 2006.

[22] J. Yang, J. Wang, M. Clements, J. A. Pouwelse, A. P. de Vries, and M. Reinders. An epidemic-based p2p recommender system. In *Proceedings of the 1st Workshop on Large-Scale Distributed Systems for Information Retrieval (LSDS-IR 2007)*, pages 11–15, 2007.

Digital Object Identifier: 10.4108/ICST.COLLABORATECOM2009.8278
http://dx.doi.org/10.4108/ICST.COLLABORATECOM2009.8278