Better control on recommender systems

Esma Aïmeur * and Flavien Serge Mani Onana
Université de Montréal, Département IRO
CP 6128, Succ. Centre-Ville, Montréal (QC), H3C 3J7 Canada
Email: {aimeur, manionaf}@iro.umontreal.ca

Abstract

In the context of electronic commerce, recommender systems enable merchants to assist customers in finding available products that will best satisfy their need. However, a recommender system usually operates as a kind of black box *from which customers receive recommendations for products. Of particular interest are recommender systems based on* Collaborative Filtering*, in which customers provide the recommender system with ratings on products and receive recommendations based on the similarity paradigm. In this paper, we introduce an approach in which customers* self-control *the collaborative filtering process. More precisely, our approach uses a list of contacts, which allows a better control on the recommendations provided locally (using the list of contacts), while still being able to access and even influence the global recommendations of the system (using the whole database of customers). We believe that our system allows more confidence in the customer because it enables him to maintain real-time statistics on his similarity with customers forming his ring of contacts.*

1. Introduction

The proliferation of online stores as an interactive mean for proposing information and products to consumers drives a high need for selective choices. Indeed, it becomes primordial to assist customers in their choices for a specific product (CDs, books, movies, etc.). Such assistance is supported by a recommender system whose role is to provide the consumer with a selection of products that better suit his preferences and requirements. Recommender systems have been widely used in the past and they are now being considered a core part of electronic commerce applications to increase customer satisfaction and confidence, thus encouraging them to make further purchases. They have been identified as a research area on its own in the mid-1990s. The term "recommender system" was coined by Resnick and Varian [16] as a generic replacement for "*collaborative filtering*", a phrase proposed earlier by Goldberg *et al.* [7], the developers of the first recommender system.

Recommender systems use several algorithms to efficiently output product recommendations to customers. For this purpose, the customer usually provides the recommender system with data such as the characteristics of the product he is looking for, ratings, demographic data (e.g. age, sex, annual income, address), etc. The recommender system then applies one or several filtering techniques (Section 2) and outputs product recommendations. From the customer's point of view, the recommender system is a kind of black box from which he receives a list of products that could satisfy his need or interest. For instance, in the case of collaborative filtering (Section 2), the recommendation for a customer is based on his similarity in terms of ratings with other customers that constitute what we shall call his *neighbourhood*. The customer doesn't really know his *neighbours*. The most important thing for collaborative filtering is to take into account the products that have been rated by both the target customer and the other customers, even if the target customer does not know who are the customers that influence, by their ratings, the products recommended to him.

The purpose of this paper is to provide a recommender system in which each customer manages his own neighbourhood consisting of a list of contacts. A *contact* for a customer, U, is another customer that U has chosen, according to the similarity of their ratings on some products. Our task is to help U maintain his list of contacts in such a way that, at any moment, he knows how and by which factor the ratings of each contact influence the recommendation. Also, contrary to related work on recommender systems based on trust or Web of trust (Section 4.1) and in which the overall system is influenced, we provide the customer with local and global recommendations of products. Local recommendations restrict the neighbourhood of the customer to his list of contacts, while this neighbourhood contains all the customers of the system in the case of global recommenda-

* Research supported in parts by Canada's NSERC.

Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)
0-7695-2511-3/06 $20.00 © 2006 IEEE

tions. Advantages of using simultaneously local and global recommendations are given in Section 5.2.

Furthermore, we provide U with a set of operations that allow him to view *statistics* on each of his contacts. Here, operations are related for instance to *quarantine*[1] a contact, *delete* a contact, view the graph of similarity for a given contact, etc. The idea of statistics constitutes the main originality of our approach, in the sense that it brings the control of the recommender system at the customer's level. More precisely, the statistics enable the customer to know those of his contacts that are getting closer to him, as well as those that are getting farther away from him, after interacting with the system. The customer therefore uses the set of operations as a toolbox that helps him to make decisions with respect to a given contact.

We review in Section 2 the basic notions of recommender systems. In Section 3, we describe our concept of statistical information management. We review some related work in Section 4. The description of our system, with experiment and evaluation, is given in Section 5. This is followed by a discussion and conclusions with directions for future work (Section 6).

2. Recommender systems

The current interest in recommender systems is justified by the plethora of applications dealing with information overload in order to provide personalized content and services to customers. Examples of such applications are electronic commerce, digital libraries and knowledge management. A recommender system relies on the product's features and/or previous customer ratings in order to provide an opinion or a list of selected products that assists the customer in evaluating products that are not yet rated by him. Based on how recommendations are made, five main types of recommendation techniques are identified [4]: *Content-based Filtering*, *Collaborative Filtering*, *Demographic Filtering*, *Utility-based Filtering* and *Knowledge-based Filtering*.

In this paper, we introduce our approach to self-controlling recommendation of products by concentrating on collaborative filtering.

Collaborative Filtering (CF) was first introduced by the developers of Tapestry [7], a filter-based electronic mail system designed for the intranet at the Xerox Parc Palo Alto Research Center. The main characteristic of CF is that it accumulates the customer's ratings of products, identifies customers with common ratings and offers recommendations based on inter-customer comparison. In other words, recommendations for a given customer are based on the behaviour and the evaluations of the other customers [15].

[1] In this paper, quarantine means suspending for a while.

There are three main categories of CF algorithms [3]: *memory-based* algorithms, *model-based* algorithms and *hybrid* algorithms. Memory-based algorithms use statistical techniques on the entire database of votes to find the neighbourhood of customers having similar tastes. This neighbourhood is then used to output recommendations. Model-based algorithms use a probabilistic approach to produce predictions, which can include two algorithms in machine learning: the cluster model and the Bayesian network model. Hybrid CF algorithms [13] use both memory-based and model-based algorithms.

We shall use a popular (memory-based) CF technique, known as the k-*nearest neighbours*-based (kNN-based) algorithm [6, 5]. Here, data is represented by a matrix whose entry $v_{u,i}$ represents the rating customer u gave to product i. This entry is set to a null value in case customer u has not rated product i, in which case this entry is not used in the computations. Suppose that the merchant's catalogue, T, contains n products, $p_1, p_2, \cdots, p_n$, and that t customers, $u_1, u_2, \cdots, u_t$ have rated some products from T. Rating predictions for a given customer are produced in two stages, as we now review.

First, the *Pearson correlation* [15] is used to compute similarity between customers, using equation 1. The results obtained range from -1 for negative correlation to $+1$ for perfect positive correlation. Specifically, let

$$cor(c,u) = \frac{\sum_{j\in J}(v_{c,j} - \overline{v_c})(v_{u,j} - \overline{v_u})}{\sqrt{\sum_{j\in J}(v_{c,j} - \overline{v_c})^2 \sum_{j\in J}(v_{u,j} - \overline{v_u})^2}} \quad (1)$$

stand for the correlation between customers c and u, where J is the set of products rated by both customers c and u, $v_{i,j}$ is the rating customer i gave to product j and $\overline{v_i}$ is the average rating of customer i for the products that belong to J, for $i \in \{c, u\}$.

The kNN equation 2 is then used to predict the rating of a customer c on a product j. As proposed in [3, 19], the resulting predictions are sorted and the k of them with the highest values are considered for recommendation purposes:

$$P_{c,j} = \overline{v_c} + \frac{\sum_{u\in U} cor(c,u)(v_{u,j} - \overline{v_u})}{\sum_{u\in U} |cor(c,u)|}, \quad (2)$$

where U is the set of all the customers who have supplied a rating for product j. (In particular, customer c, for whom predictions are being computed, does not belong to U).

In practice, none of the filtering techniques presented above are perfect.

Limitations of CF. The CF technique is, unfortunately, not perfect and it suffers from a set of weaknesses. The main one is *data sparseness*; in real applications, a rating matrix comprises millions of products and millions of customers. However, a typical customer will rate only a small number of products. As a consequence, the rating

Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)
0-7695-2511-3/06 $20.00 © 2006 IEEE

matrix presents many empty cells. This is a problem since customer similarity—using the Pearson coefficient—is not computable in most cases. The second weakness is *cold start*: when new customers have not yet expressed any rating, it is difficult to provide a recommendation based on the profile similarity. As an intuitive approach to alleviate these limitations, hybrid recommender systems [4]—not to be confused with hybrid CF algorithms (see above)—have been widely used to take advantage of the strengths of each scheme while avoiding their weaknesses. A third weakness is the vulnerability of recommender systems to attacks. Driven by commercial, personal or political motives, this problem, known as *shilling* [10], is caused by fake customers whose purpose is to influence customer choices by providing false ratings. Two scenarios of shilling immediately come to mind. First, one in which malicious manufacturers would shill the recommender system by posing as customers in order to unfairly push the sales of their products. In a similar fashion, competitor merchants could mangle the recommender system so the quality of recommendations decreases and therefore increases customer dissatisfaction. Another limitation concerns merchants who could create fake customers and provide ratings for them, modify existing ratings or even falsify their recommendation algorithms, with in their mind the same objectives as those of customers that practice shilling. We shall call this limitation a *boosting* attack. The difference between shilling and boosting attacks is that the former is external to the system while the later is internal and, therefore, more difficult to be controlled by external customers.

Our approach opens another way to circumvent the limitations presented above (Section 5.5).

3. Statistical information management

Recommender systems are very useful, but they usually produce recommendations without allowing much control to the customer on the recommendation process. At best, he can just express his feedback to the system by rating the products that have been recommended to him.

In the context of CF, how can a customer have control on a recommendation process that uses his neighbourhood? In other words, how can a customer decide which customers may be included in a CF process from which he will receive product recommendations? These questions are partially answered by the use of a Web of trust (see for example [12]). Here, customers express the trust (Section 4.1) they have about a given customer through votes. These votes are used during the recommendation process and affect its output. However, expressing trust as votes is not sufficient. For example, it is important to highlight some aspects in which a given customer ratings may be useful, even if they are not in other aspects. Let us illustrate this in the movie domain. Given two customers, U and V, U may trust V for some genres of movies (e.g. action) while not trusting him for others (e.g. comedies). In this case, the vote that U expresses about V is more or less subjective.

From our point of view, one way to answer the above questions is to provide the customer, U, with a mechanism that allows him to supervise the evolution of his similarity with customers that take part in recommendation process. According to U, the evolution of the similarity is about the storage and maintenance of statistics on how much a given customer comes closer to or goes farther from U, after at least one of them has interacted with the system: we call this *Statistical Information Management* (SIM).

The customer U can apply SIM on a small set of customers previously chosen by U. In this paper, chosen customers are called *contacts* of U and the set of these customers is the *list of contacts* of U. Therefore, a recommendation process that uses a list of contacts is considered to be *local*. This is opposed to the *global* recommendation process, which does not restrict the customers that take part in a recommendation process to be contacts of U.

SIM also consists of a set of operations that U can apply on his contacts. For example, U may want to *delete* a given customer from his list of contacts, *quarantine* a contact for a while or *restrict* the contact in participating in the recommendation process for certain types of products. For the last point, U uses statistical information to analyse the evolution of his similarity with a given customer, by considering two cases:

– General statistics: They give an overview of the evolution of the similarity between U and a given customer for all types of products. In the case of movies, for instance, U may only need to know if he is *continuously similar* (with respect to a certain *precision* that U selects) to a given customer.

– Partial statistics: They present the evolution of the similarity between U and a given customer for each type of product. In the case of movies, U may have an interest on how similar he is with a contact, according to action movies, comedies, etc.

Our task is to provide a recommender system that permit the customers to monitor the recommendation process. With this in mind, we identify two levels of credit that customers may have about other customers. Here, the term "credit" is used to quantify trust that a customer has about each contact in his list or trust that customers of the whole system have about a particular customer. The two levels of credit are supposed to take into account the fact that different customers may have different monitoring approaches.

– Global *credit* for a customer: All the customers of the system express the credit that they grant to a given customer. More precisely, this credit is based on trust the customers have in this customer (Section 5.4).

Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)
0-7695-2511-3/06 $20.00 © 2006 IEEE

– Local *credit* of a customer: It refers to the credit that a particular customer has for each customer on his list of contacts (Section 5.4).

In fact, the credit is updated through the set of operations used in SIM. Adding a new contact increases the global credit, while deletion or quarantine decreases that credit. As for the local credit, U can initialise it with the global credit for the customer if he decides to add that customer to his list of contacts. Then, U manages the local credit of that contact according to the evolution of similarity with him. This means that after interacting with the system, depending on how much and in which direction (increasing or decreasing) the similarity between U and his contact has changed, the local credit will remain unchanged, increase or decrease. More detail about this is given in Section 5.4.

The idea of managing contacts may also alleviate shilling attacks [10]. In fact, customers have control and manage their lists of contacts. The recommender system can therefore use the evolution of the global credit to discover shilling attacks (Section 4.2).

Furthermore, our solution is a social network [17] in which a customer can access statistical information about himself: global credit, local credit, how many times he is added, quarantined or deleted, in which types of products he is well accepted by other customers, etc. He can also access statistical information about other customers.

4. Related Work

4.1. Trust

Trust has its roots in human interactions as a major element of cooperative endeavours. Researchers from different disciplines have highlighted the importance of trust in their own domain. As a consequence, different definitions of trust are continually proposed in the literature according to the specific perspectives of each discipline. We retain the definition according to which trust is *the quantified belief by a trustor with respect to the competence, honesty, security and dependability of a trustee within a specified context* [8].

Trust and Recommendations. In recommender systems in which high-quality recommendations can be obtained using collaborative filtering (Section 2), recommendations are traditionally generated for a target customer by drawing on the rating history of a set of suitable recommendation partners [12]. However, a partner can be identified as a potential recommender based on the customer-customer similarity but this does not mean that this partner is a reliable predictor for a given product or set of products. Indeed, based on the fact that *people prefer receiving recommendations from people they know and trust* [18], trust represents a primordial factor for validating the quality of the ratings. In practice, an estimation of trust is incorporated into the similarity function used to derive a recommendation, in order to produce recommendations with the highest accuracy. Some research efforts have been made in this direction.

Web of Trust. The concept of *Web of trust* has been first introduced in OpenPGP-compatible systems [14] as a trust model for distributed systems. It is used as a replacement for central or hierarchical signing authorities in order to establish the validity of public keys based on a network of trust. In the same manner, in recommender systems, a network of trust representing trust degrees can be built [11]. Depending on how trust values are computed, the literature distinguishes between two types of trust metrics.

– Global trust metric: Given a population of customers, a global trust metric predicts a degree of trust on a specific customer A based on the appreciation of the whole population. As a consequence, the degree of trust on customer A will be the same for all the other customers.

– Local trust metric: In contrast, local trust metrics predict for a target customer different trust degrees for different customers.

Our global credit and local credit (Section 3) are new metrics that we introduce for global trust and local trust, respectively. Massa and Bhattacharjee [11] assert that trust derived from the *Web of trust* of each customer could be adopted as a second approach to alleviate the weaknesses of recommender systems discussed earlier. Based on the `Epinions.com` dataset, the authors compare the computability of two quantities that can be used as weights for every customer, namely, the widely used Pearson coefficient and Trust. They show that the second one is computable about many more customers than the first, especially for new customers, thus alleviating the problem of cold start. This contribution is extended in [2] where a "Moleskiing" recommender system is developed. The goal of this system is to make ski mountaineering safer by relying on customer opinions about the snow conditions of the different ski routes. The metric used in Moleskiing is local: it considers the personal views of every single customer and can possibly predict a trust value in another target customer that is different for every source customer.

Note that in our approach described in Section 3 and detailled in Section 5, we do not permit propagation of the credit. However, this credit can be consulted as statistical information by any customer, who may now decide to apply some actions, depending on his feeling.

In addition, O'Donovan and Smyth in [12] present two computational trust model known as profile-level trust and item-level trust and show how these trust models can lead to improved predictive accuracy during recommendation. The authors also describe a number of ways in which the trust metrics they have introduced might be incorporated into a standard collaborative filtering algorithm and evaluated each one of them against a tried-and-test benchmark

approach and on a standard dataset. They found that the use of trust values reduces prediction error rates compared to the benchmark. The aim of their work is similar to ours, however, in our case we separate our credit paradigm from the recommender algorithm. More precisely, the credit is used to make decisions on whether or not a particular contact may take part in the recommendation process. In particular, decisions may be based on the kind of products to be recommended, so that the credit about a customer could be welcome for certain products and not for others.

4.2. Shilling

In order to be effective, recommender systems are required to be open to raters without any authentication procedure. This approach, however, encourages malicious customers to cheat the system with false ratings. Many motives can be behind this kind of behaviour, among them are *fun and profit* as stated by Lam and Riedl [10]. Indeed, some well-known e-commerce entities have already admitted that their recommender systems were subject to shilling. Shilling brings out a dangerous aspect for the online store since consumers may lose their trust in commercial Web sites. In particular, incidents caused by shilling may seriously affect the reputation of the online store and reduce the number of purchases.

Trust constitutes a promising approach for overcoming attacks on recommender systems. It allows receiving recommendations from trusted customers (at a given degree). Our approach, that we will discuss in more detail in Section 5, uses a new scheme for computing recommendations based on a local trust metric or *local credit computation*, while influencing the recommender system with a global trust metric or *global credit computation* (Section 5.4). Therefore, this new scheme is intended to reduce shilling effects and still provide effective recommendations.

5. Our system

In this section, we present the architecture and components of our system. We then give an overview of our system and present the recommendation algorithms, as well as the statistics computation. We end with the description of the experiment and evaluation.

5.1. Architecture and components

Our system is designed to ensure confidence and trust in product recommendation by allowing the customer to have some control on other customers who influence the recommendation process, through their ratings. The main idea behind our system is the integration of the statistical information management paradigm within recommender systems based on a collaborative filtering technique. The statistical information enables customers to build an implicit social network, which is proved to be a good way to express trust between them. The main goal of our system is to offer a platform in which customers are their own recommendation supervisors.

Contrary to the centralized approach, in which the *server* of the recommender system functions as a black box, we introduce a *logically* decentralized architecture in which the server of the recommender system plays the role of a social broker and matchmaking facilitator between customers. The logical approach enables customers to choose and keep their contacts in the merchant platform and manage them as if they were in their own platforms. We also discuss the physical (as opposed to logical) approach in Section 5.6, to enable customers to hold a local database of their contacts as well as their own recommender algorithms. Figure 1 illustrates an overview of the logical approach, with the following components:

– Customer database: consists of the customer profiles and additional information such as global and local credits (Section 3). In particular, the customer database stores the list of contacts for each customer.

– Product database: contains the catalogue of products sold by the merchant.

– Rating database: stores the ratings provided by customers on products that the merchant has for sale.

– Recommender System: it is a three-party recommender system:

Contact recommender: recommends potential customers to form the list of contacts of the current customer.

Global recommender: is about products recommended by taking into account all the customers in the system.

Local recommender: is concerned with products recommended by limiting the recommendation process only on customers present on the list of contacts of the target customer.

Detail about these three parts of the recommender system is given in Section 5.3.

The above components constitute the merchant platform.

5.2. System overview

A new customer, U, must first register and log into the system. He is then invited to rate some popular products. He may also search by himself for products to rate on a *one-to-five* scale. Once U has rated some products[2], collaborative filtering is used to recommend customers that could be his contacts, according to the similarity between his ratings and theirs (Section 5.3). The recommendation of customers is refined as the customer rates more products. From the recommended customers, U may select some of them to

[2] At least three in our case, but this number can be modified as needed.

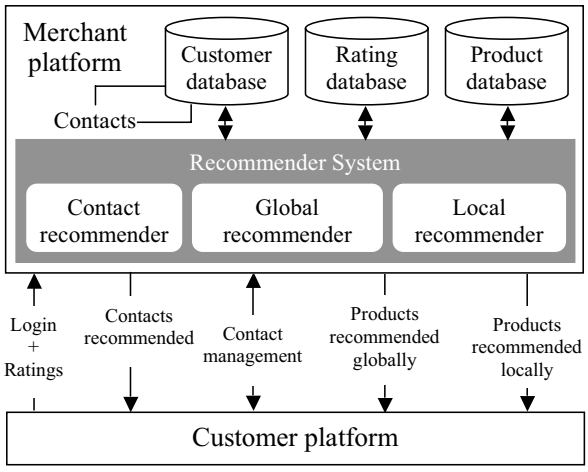

Figure 1. Architecture of the system

add to his list of contacts. At anytime, U may add, quarantine or remove contacts from his list. He may also access the profile of a given contact in order to view statistical information about him.

The local recommender applies a CF technique to compute the similarity (Equation 1, Section 2) between U and his contacts, and generate predictions (Equation 2), which provide U with local recommended products (Section 5.3). Customer U can later rate the recommended products that he has selected. For instance, in the case of movies, U may receive r recommended movies and select $t \leq r$ of them. He therefore rates the t movies he selected. Once such ratings are provided, the contact management procedure is performed to update the statistics of each contact. The purpose of this update is to inform U on the evolution of the similarity with each contact (Section 5.3). The particularity here is that the customer can include or exclude contacts from being taken into account in the recommendation process. By doing so, he decides which contacts he trusts more, for instance, for a certain category of products. For example, if movies are products to be recommended, the customer may decide to include some contacts when looking for horror movies, while he may exclude them in the case of comedies.

As for the global recommender, it applies a CF technique to compute the similarity between U and any customer with whom he has rated at least one common product. This recommender generates predictions and provides U with global recommended products (Section 5.3).

Our system is called a *three-party* recommender system because of its possibility to output recommended contacts and, simultaneously, local and global recommendation of products, so that the customer can choose a recommended product from either of them. In particular, the local versus global approach has some advantages, such as:

– It provides the customer with real time feedback about products that interest the whole set of customers compared with products recommended from his list of contacts.
– The customer is able to evaluate directly the quality of his list of contacts. More precisely, depending on how much he is satisfied by the global recommendations on the one hand, and local recommendations on the other hand, he could decide to modify his list of contacts. In this case, he runs the contact recommender process (Section 5.3).
– Furthermore, the local versus global paradigm helps the merchant compare the global recommender versus the local one. The merchant can then improve the quality of the ratings provided by his customer(s). For example, if local recommendations are better that global ones, the merchant could find the global credit of his customers and additional statistical information such as the number of times a given customer has been deleted by others. This can enable the merchant to quarantine a customer if he notes that this one does not have any more credit from other customers. As in the case of the local recommender, the global recommender does not take into account the ratings of the customers that are quarantined.

Moreover, the customer can access the profile of other customers and view the number of products they have rated, the number of contacts they have, the number of customers that have chosen him as a contact, the number of times he has been deleted or quarantined, as well as the similarity of ratings with these customers. This statistical information provides the customer with more confidence in choosing his contacts based on *global* and *local credit* information.

5.3. Three-party recommendation algorithms

In our system, we apply the CF technique to both global customers and local contacts, to output global and local recommendations, respectively. We also use the CF technique to compute the similarity between customers and output contact recommendations, as described in the following matchmaking algorithm.

Social matchmaking algorithm. The social matchmaking algorithm provides a mean to build links between customers by recommending contacts to them. Contact recommendations are obtained from the similarity of ratings between customers. The recommendation process is based on the collaborative filtering technique. Customers who present greater similarity with the current customer are recommended. A minimum of three products rated in common is required before a contact could be recommended. Given a customer U, the social matchmaking algorithm proceeds as follows: For each customer A who has rated at least three products in common with U, the similarity between U and A is computed, using for example Equation 1 of Section 2.

Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)
0-7695-2511-3/06 $20.00 © 2006 IEEE

Customers are then sorted in decreasing order of the similarity value, and those with the highest positive similarity value are presented to U.

Global recommendations. The global recommendation of products uses the rating matrix as input. Given a customer U, recommendations are generated as follow: U chooses a minimum credit, c_{min}, that may be taken into account during the recommendation process. For each customer A from the customer database who has global credit greater than c_{min} and who has rated at least three products in common with U, the recommender system computes the similarity between U and A (Equation 1 has been used in our case). For each product j that the customer has not yet rated, the recommender system computes the predicted rating, $P_{U,j}$, of U on j (Equation 2 has been used). The recommender system then outputs products j with the highest predictions $P_{U,j}$ (the top k predictions have been used).

Local recommendations. The local recommendation of products is similar to global recommendation. The difference is only about the choice of customers from the list of contacts of U, instead of the whole customer database.

Both local and global recommendations are generated simultaneously and presented to the customer in a two-column web page.

5.4. Statistics computation

Let n be the total number of customers in the system and G the total credit for these n customers.

Global credit computation. Customer U, with global credit u_g, can only influence the credit of his contacts. Suppose A is a contact of U, with global credit a_g. We define the global *add/delete credit unit*, λ_G, as the maximum value that all the customers of the system can together add to or subtract from the global credit, for any customer who is added to or deleted from a list of contact. We define the global *quarantine/de-quarantine credit unit*, γ_G, similarly, in the case that the customer is quarantined in a list of contacts. Here are operations used by U to update global credit of customer A: *add*, *delete*, *quarantine* and *de-quarantine* a customer from/to a list of contacts:

add: $a_g = a_g + \frac{u_g}{G}\lambda_G$ ***quarantine***: $a_g = a_g - \frac{u_g}{G}\gamma_G$

delete: $a_g = a_g - \frac{u_g}{G}\lambda_G$ ***de-quarantine***: $a_g = a_g + \frac{u_g}{G}\gamma_G$.

One can consider the quarantine operations as a proportion of the delete operations. In our case, we take $\gamma_G = \frac{\lambda_G}{2}$, meaning that quarantine is a semi-deletion and, similarly, de-quarantine is a semi-addition. Figure 2 presents the profile of "movie_critic", who is a contact of "user01". His global credit is 83. Suppose that $\lambda_G = 1$, then $\gamma_G = \frac{\lambda_G}{2} = \frac{1}{2}$. Therefore, the global credit of "movie_critic" will decrease by 1 or $\frac{1}{2}$, depending on whether "user01" deletes or quarantines him from his list of contacts. The same result will occur considering any other customer who has "movie_critic" in his list of contacts. Also, if "movie_critic" is added or de-quarantined from any list of contacts, his global credit will increase by 1 or $\frac{1}{2}$, respectively.

Movie_critic profile

Local credit	31
Number of ratings	45
Number of joint contacts	6
Rating similarity	0.63

Global credit	83
Number of ratings	122
Number of contacts	52
Contact of (customers)	13
Customer has been rejected	3 times

Figure 2. Statistical information about "movie_critic" (as seen by "user01").

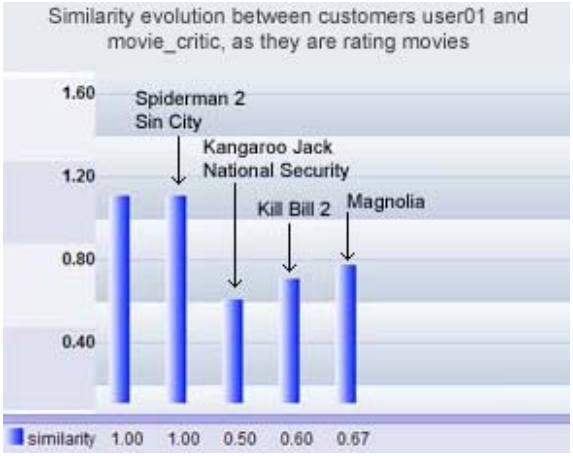

Figure 3. Statistical evolution of local similarity

Local credit computation. Let us take a contact A, with local credit a_ℓ. Local credit is computed similarly to global credit. However, since U is the supervisor, we only deal with the local *delete/add credit unit*, λ_L, and local *quarantine/de-quarantine credit unit*, γ_L. Here are operations to update local credit:

add: $a_\ell = a_\ell + \lambda_L$ ***quarantine***: $a_\ell = a_\ell - \gamma_L$

delete: $a_\ell = a_\ell - \lambda_L$ ***de-quarantine***: $a_\ell = a_\ell + \gamma_L$.

In Figure 2, "movie_critic" has 31 as local credit. This is updated in a way similar to that of global credit (see above).

Similarity evolution. We enable the customer to view the evolution of his similarity with another customer, in both the global and local points of view. Figure 3 illustrates such evolution between customer *user01* and customer *movie_critic* in the movie domain, as we chose that domain to implement our system (Section 5.6). Movies that

deeply influence similarity between the two customers are highlighted (*Spiderman 2*, *Kangaroo Jack*, etc.).

5.5. Solving some limitations

Our system opens up a way to limit the weaknesses presented in Section 2: For *sparseness*, the list of contacts restricts local recommendations to customers that have ratings similar to the owner of the list. Therefore, the *restriction* of the rating matrix to those customers presents less empty cells than the whole one. *Cold start* is resolved in our case since a new customer needs to rate some popular products (or products of his choice)—but these products must exist in the merchant catalogue—before the recommendation process starts. As for *shilling*, the statistical information management, based on contacts, allows more vigilance in discovering customers who may cheat the system by providing false ratings. Each customer controls his contacts through the similarity evolution and can therefore decide to delete or quarantine any contact that does not maintain a stable similarity with him.

Our system also limits boosting attacks from merchants themselves if a physically decentralized approach (Section 5.6) is used. In this physical approach, customers manage contacts in their own platforms and have more control on the recommendation process than in the logical case (this is discussed in Section 5.6).

5.6. Implementation

We implemented our system with a free open source LAMP technology, using Linux, Apache, MySQL and PHP. This implementation was about movie recommendations. Our system does not gather personal information on customers, only a username (pseudonym) and a password are needed.

The implementation has been done for the logically decentralized approach (Section 3), in which the customer's list of contacts is placed within the merchant's platform. There are two main advantages to this approach when compared with one in which the customer would download the list of contacts in his own platform: on the one hand, the merchant can retain physical control over the customers and their ratings, which should be his property by right. On the other hand, this approach adds confidence in the privacy of the merchant's data, such as his customers and their associated ratings, which is manipulated within the merchant's platform [1].

The problem with the logically decentralized approach is that the merchant still has access to all the customer profiles, as well as their list of contacts, and several problems could arise from that. For instance, what could be done if the merchant decides to shill his own system, by creating and managing customers with well-chosen ratings? It would therefore be desirable to evolve in a physically decentralized approach in which customers really control the recommendation process. The main problem here is the risk of allowing a customer to get information about other customers. But this doesn't really matter for the following two reasons: *first*, recommender systems that use trust usually give away information about customers; for instance, crawling can be used as it has been the case in this paper or in [11]. The *second* reason is that in our case, customers do not provide the system with personal information: only a pseudonym is required. Hence, our physical approach will not open up a way to privacy violation. Moreover, this approach presents two important advantages:

– The customer's control on the recommendation process would be improved since the customer could choose his own recommendation algorithms, hopefully different from those used by the merchant. There would be a two-party computation in which the customer's input consists of his recommendation algorithm and his list of contacts, whereas the merchant's input is his catalogue of products. The output of the computation is the list of products recommended to the customer. Today, there are several recommendation algorithms, but a given merchant typically uses few of them. Allowing customers to use their own algorithms[3] may therefore open new issues for recommender systems.

– The customer could add customers that do not exist in the customer database within the merchant platforms. Such customers could, for instance, come from other merchants' platform or even from his acquaintances (friends, showbiz celebrities, etc.). The most important thing in this case is that these customers may have rated some products present in the merchant's catalogue. Also, acquaintances are managed locally, meaning that they do not have global credit—at least, as long as they are not customers (of the merchant). It is important to note that this leads to combining word-of-mouth advice about products with ratings issued by merchant's customers.

We shall detail the issues related to the physical implementation in future work. For the moment, we describe the tests and validation that we performed, based on the logical implementation.

5.7. Tests and validation

Dataset. The test and validation dataset, consisting of 6638 users, was extracted from Epinion's website (`epinion.com`). Epinion is a popular community based portal, where customers can rate all kind of products, from electronics to movies. Ratings are expressed in a one-to-five scale. Moreover, customers have the possibility to choose other

[3]The customer may create or download (from the Internet) such algorithms.

Table 1. Epinion dataset before and after the cleaning process.

	Original set	Cleaned set
customers	6638	426
movies	11413	6776
ratings	49429	15781
friends	64231	17027
	Ratio	**Ratio**
ratings/movie	4.33	2.33
ratings/customer	7.45	37.04
friends/customer	9.68	39.97

customers and add them to their *Web of Trust*. A customer's Epinion Web of Trust is simply a set of self selected trusted customers. We developed a *crawler* that fetched data about Epinion customers, friends and movie ratings using the *screen scraping* technique.The dataset includes essentially a set of customers' movie ratings, and customers' friend networks. We conducted a crawling similar to that used in [11], but limited our dataset to movie ratings. The dataset obtained showed that 93% of customers have rated less than 5 movies and have 5 friends. We decided to clean the Epinion's grabbed dataset and keep only customers who rated five movies or more, and have at least five friends. Table 1 shows the dataset before and after the cleaning process. This cleaning process was necessary in order to apply the evaluation protocol described in section 2.2. We need customers with a minimum of ratings and friends in order to generate significant recommendations.

Evaluation protocol and metric. We conducted a 5-fold cross validation on the epinion movie ratings dataset. The 5-fold cross validation consists in randomly subdividing each customer's ratings into 5 subsets. Iteratively, each subset (20% of customer's ratings) is selected as the evaluation set, and training is performed on the 80% remaining ratings. Recommendations are generated based on training sets, then we compare these recommendations with the evaluation set. Recommendations are evaluated using the Mean Absolute Error (MAE) as a metric. MAE is the mean deviation between rating recommendations and their true customer-specified values. MAE has been used in many recommender systems evaluation studies [9], because it is easy to implement and interpret. It is computed as follows:

$$MAE = \frac{\sum_{i=1}^{n} |r_i - e_i|}{n} \quad (3)$$

where n is the number of recommended movies that are in the evaluation set, r_i is the recommendation rating for the i^{th} movie, e_i is the actual customer rating for the i^{th} movie. The lower the MAE value (closer to zero) the better is the recommendation accuracy.

The 5-fold cross validation protocol was conducted on two different experiments. First, we considered global recommendations. Then, we repeated the same experiment for local recommendations. Our goal is to show that local recommendations are more accurate than global recommendations.

Results. Figure 4 shows the global MAE distribution resulting from the global recommendation process. The trendline is a 3 degree polynomial approximation of the distribution. The graph shows that the MAE goes smaller as customers rate more movies. This is not a surprising result, since recommendations accuracy is a positive function of the number of ratings provided by customers. In this case, MAE equals 1.15, meaning that the recommendations were erroneous by 1.15 deviation from actual customer ratings.

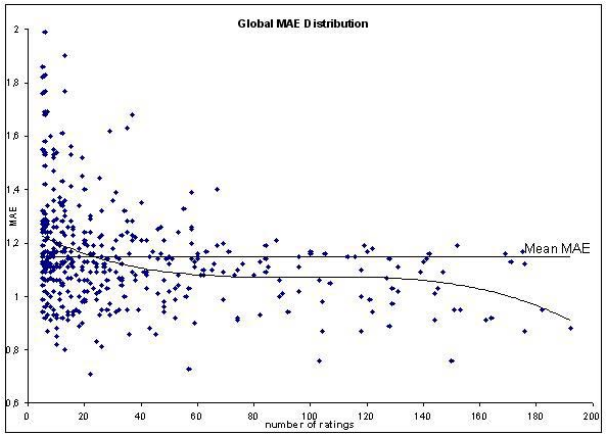

Figure 4. Global MAE distribution function of customers' number of ratings

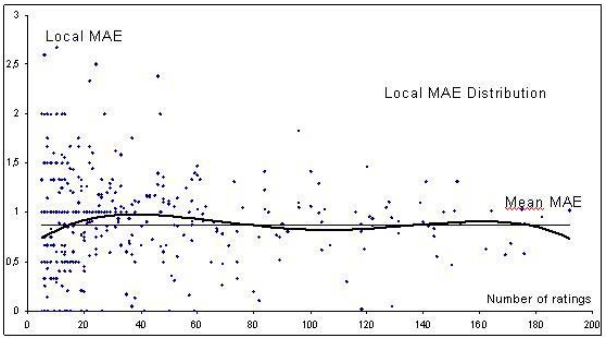

Figure 5. Local MAE distribution function of customer's number of ratings

Figure 5 shows the same results as Figure 4, however, in the case of local recommendations. Here, MAE equals 0.85, which is better than that of the global recommendations. However the 3 degree polynomial approximation does not

Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)
0-7695-2511-3/06 $20.00 © 2006 IEEE

show substantial improvement in recommendations while the number of ratings increases. The accuracy of recommendations does not seem to be positively correlated with the number of contacts. The main reason behind these results is that increasing the number of ratings while keeping a small set of contacts would not result in better local recommendations. The same reasoning is valid when one increases the number of his contacts, while keeping a small set of movie ratings.

6. Conclusions

In this paper, we have proposed local and global recommendation paradigms to enable customers to self-supervize the recommendation of products in electronic commerce. For that purpose, we have introduced the notions of statistical information management and of a list of contacts for a given customer. We have also defined a set of operations (add, delete, (de-)quarantine) for creating and managing contacts. These operations are related to the computation of credits, which we use to express trust concerning contacts. For this purpose, we have introduced the global credit for a customer, which is the credit that the whole set of customers who belong to the recommender system give him, and the local credit, which is the credit that a given customer has for each customer in his list of contacts.

Moreover, we have enabled a given customer to view the evolution graph of his similarity with any of his contacts and, consequently, make decisions depending on whether a given contact comes closer to or moves farther from him.

The results of this logically decentralized implementation show that local recommendations are better than global recommendations. Therefore, our system provides customers with more control and confidence on the recommendation process. We leave the implementation of the physically decentralized approach for further research.

Acknowledgements

We are most grateful to Houssein Ben-Ameur, Gilles Brassard, Alexandre Desmarais-Frantz and Sébastien Gambs for useful discussions.

References

[1] E. Aïmeur, G. Brassard, J. M. Fernandez, and F. S. Mani Onana. Privacy-preserving demographic filtering. In *Proceedings of the 21st Annual ACM Symposium on Applied Computing*. Dijon, France, 2006. To appear.

[2] P. Avesani, P. Massa, and R. Tiella. A trust-enhanced recommender system application: Moleskiing. In *Proceedings of the 20th Annual ACM Symposium on Applied Computing*, pages 1589–1593. Santa Fe, New Mexico, 2005.

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

[4] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 4(12):331–370, 2002.

[5] T. M. Cover. Rates of convergence for nearest neighbor procedures. In *Proceedings of Hawaii International Conference on System Science*, pages 413–415, Hawaii, 1968.

[6] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transaction on Information Theory*, IT-13:21–27, 1967.

[7] T. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[8] T. Grandison and M. Sloman. Trust management tools for Internet applications. In *First International Conference on Trust Management*, pages 28–30, Heraklion, Greece, 2003.

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

[10] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th World Wide Web Conference*, pages 254–255, New York, New York, 2004.

[11] P. Massa and B. Bhattacharjee. Using trust in recommender systems: An experimental analysis. In *Proceedings of 2nd International Conference on Trust Management*, pages 221–235, Oxford, U.K., 2004.

[12] J. O'Donovan and B. Smyth. Trust in recommender systems. In *IUI'05: Proceedings of the 10th International Conference on Intelligent User Interfaces*, pages 167–174, San Diego, California, 2005.

[13] D. Pennock, E.Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI 2000*, pages 473–480, Stanford, California, 2000.

[14] PGP. Explanation of the web of trust of PGP. `http://www.rubin.ch/pgp/weboftrust.en.html`. Accessed 16 December 2005.

[15] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 Computer Suported Collaborative Work Conference*, pages 175–186, Chapel Hill, North Carolina, 1994.

[16] P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[17] J. Scott. *Social Network Analysis: A Handbook*. SAGE publications, second edition, 2005.

[18] R. Sinha and K. Swearingen. Comparing recommendations made by online systems and friends. In *Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*, Dublin, Ireland, 2001.

[19] E. Vozalis and K. G. Margaritis. Analysis of recommender systems' algorithms. In *Proceedings of the 6th Hellenic European Conference on Computer Mathematics and its Applications (HERCMA-2003)*, Athens, Greece, 2003.