Research on the Personalization Recommendation of Mobile Business Based on the Ant Colony Optimization Algorithm

Jia-Lu Shen*,1, Qi-Feng Ding2

1 School of Economics and management
Tongji University, Shanghai 200092, P.R. China
2 The State Radio-monitoring-center Testing Center
Beijing 100041, P.R. China
* shenjialu_tongji@163.com

Abstract — In this paper, we prompt a new method of personalization recommendation of mobile business based on the ant colony optimization algorithm. Ant Colony Optimization (ACO) is a heuristic algorithm which has been proven a successful technique and applied to a number of combinatorial optimization (CO) problems. To solve the problems of scalability and sparsity in the collaborative filtering, this paper proposed a personalization recommendation algorithm based on rough set is proposed. The algorithm refine the user ratings data using dimensionality reduction, then uses a new similarity measure to find the target users' neighbors, and then generate recommendations. To prove our algorithm's effectiveness, the authors conduct experiments on the public dataset. Theoretical analysis and experimental results show that this method is efficient and effective.

Keywords - Personalization Recommendation; Collaborative Algorithm; Ant Colony Optimization Algorithm; Mobile Business

I. INTRODUCTION

With the rapidly increasing amount of information in the networks, there is a serious need for a new technology to help people find what they want from a huge mass of data. Personality recommendation system emerges as the times requires, which is used to help users find information they are interested in. The provided personalized service is accepted by more and more E-commerce Web site, digital library and many other similar fields, and becomes one of the most important fictions in these systems. At present, almost all large-scale e-commerce systems, such as Amazon, eBay, and taobao [1], use recommendation systems in a variety of modes.

Many researchers have proposed various kinds of CF technologies to make a quality recommendation. All of them make a recommendation based on the same data structure as user-item matrix consisting of their rating scores. CF as user based collaborative based having on the users same data and items collaborative filtering [2]. There are filtering. User based CF two methods in and item based assumes that a good way to find a certain user's interesting item is to find other users who have a similar interest. So, at first, it tries to find the user's neighbors based on user similarities and then combine the neighbor users' rating scores, which have previously been expressed, by similarity weighted averaging. And item based CF fundamentally has the same scheme with user based CF. It looks into a set of items; the target user has already rated and computes how similar they are to the target item under recommendation. After that, it also combines his previous preferences based on these item similarities [3].

The traditional collaborative filtering algorithm works by building a database of preferences for items by users [4]. To find information that the target user may probably be interested in, we first discover the target user's nearest neighbors, which are other users who have historically had similar taste to the target. The traditional nearest-neighbor collaborative filtering recommendation algorithms are used to face the challenge of extreme sparsely of user rating data [5].

To solve the difficulties of the extreme sparsity of user rating data, in this paper, we first reline the user ratings data using dimensionality reduction aiming at solving the problems of sparsity in the collaborative filtering, then uses a new similarity measure to find the target users' neighbors [6].

The experimental results show that the performance of the present item-based collaborative filtering algorithm is improved, even with extreme sparsity of data [7]. An integral recommendation system is composed by 3 parts: input module (to collect customers' information and behaviors), recommendation algorithm module and output module. Recommendation algorithm is the core of the whole system [8]. In order to recommend accurately and opportunely, many experts researched the algorithm, and proposed many different kinds of algorithms. The most popular algorithms can be divided into 3 categories:

First, it is rule-based recommendation, which is to search relative items and recommend based on associated rules. Its most important advantage is that it doesn't rely on users' explicit input, and doesn't have the problem of data sparsity. However, we need extract, clean, transform data and keep data in the classification of store from a large number of User Access Logs to establish rules, which cost a lot. What's more, it is very hard to adapt to customers' changing interests.

Second, content-based recommendation is an Item-to-Item Correlation way classified in Schafer, which is a way to recommend other similar objects based on customers'
choices. It entirely depends on the characteristics description of the items. And the recommendation is limited in some familiar hinds and areas, we cannot get cross-species recommendation. Although it is helpful to save search time, it cannot completely reflect the real value of personalized recommendation.

Collaborative filtering technology is the most popular one in recommendation systems nowadays. Its basic thought is that we should find the nearest neighbor, which is the target-user's most similar one in the light of item-ratings, forecast target-user's scores on the items they haven't marked according to their neighbor's, then sort and choose the highest to recommend to the target-user. Collaborative filtering is a user-rating-based technology.

II. THE BACKGROUND AND FRAME WORK OF PERSONALIZED MODULE

However, there are a large number of users and items in traditional E-Commerce web site, and this number does increase now. In the meantime, the mark of users is very scared. Usually, there is less than one marked by users in every hundred items. This is data sparsity of Collaborative Filtering. Stated thus, we can conclude the below Table 1. We can conclude from the table that there are two modules in recommendation systems' input in traditional E-Commerce. One is the module of users' interests which contains the browsing history, the ratings to items, and so on. The other is the module of items' information. Both of them are historical information. Recommendation algorithm is an algorithm that we sort the items based on the information of items and users' interests, and recommend the items in accordance with users' interests, which is so-called output. The whole architecture of recommendation is like Fig. (1).

Compared to traditional E-Commerce, mobile business has its own characters, such as public popularity, user particularity, position relevance, real-time, which demand more in personalization and real-time of recommendation. So, personalized recommendation of mobile business cannot completely copy from that of E-Commerce.

Because of user particularity and position relevance of mobile service, when users visit web sites, we can get their information by mobile phone numbers, and get their positions. Then, identify users' classifications by the information, and do different recommendations for different classification. Based on the architecture of E-Commerce personalized recommendation system, we propose new personalized recommendation system architecture for mobile business applications. In this architecture, users' information is divided into 2 parts, historical information and immediate information, both of which are stored separately. Users' historical information mainly contains browsing histories, ratings of items, purchasing histories, and so on. The module of users' immediate information is used to store those that reflect the immediate interests of users, such as position, time, etc. In recommendation module, both of them are used as factors to forecast users' preferences, and generate recommendations to them, then record the recommendation into the module of users' history information simultaneously, which is really helpful to avoid repeated recommendation.

The whole architecture of personalized recommendation system in mobile business is like Fig. (2).

Refine this system further; we can get a flow chart like Figure 3. The specific recommendation process can be divided into 3 steps:

First, we need to obtain users' immediate preferences. We can forecast users' immediate ratings to items according to the position or other factors that we can get immediately, and reject some items that don't meet the reality, which is helpful to improve the efficiency of searching. Position is really a very important parameter in recommendation, so the service on the basis of position is one of the most remarkable features of mobile business.

Second, it is very important to distinguish users' identities. In this step, we need judge whether the user is the registered member or not based on the theory that mobile phone number is the unique identification of our users. Third, the last step is to do recommendations. At the beginning of this step, we take different ways to recommend for users according to their different identities. If you are a newer and have no history information, the system will sort the items on the basis of immediate preferences obtained in the first step, and choose the most possible interesting items as recommendation. If position is the measure of preferences, we choose those whose position rating is the highest, which is really effective to solve the cold-start problem. Else, if you are not a newer and have ever visited our sites before, we should get your historical ratings about items and related purchasing histories from users' history profile, and cluster integrating users' immediate interests, then search the clearest neighbor. After that, we should forecast users' ratings for items from their neighbors, and sort them by ratings, then select the former n ones. In the end, some of them that have been recommended to the user before should be removed to avoid repeated recommendation, the others are recommended to the user. Meanwhile, record the recommendations into users' historical profile.

### Table 1. The Most Popular Recommendation Algorithms

<table>
<thead>
<tr>
<th>Recommendations System</th>
<th>Input</th>
<th>Criteria</th>
<th>Output</th>
<th>Algorithms</th>
<th>Drawbacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-based recommendations</td>
<td>The browsing and purchasing history of users</td>
<td>Select one or more items with the highest rank and recommend them to the user</td>
<td>User recommendations</td>
<td>Retrieval of user ratings and recommendation of user ranked</td>
<td></td>
</tr>
<tr>
<td>Item-based recommendations</td>
<td>The characteristics of items</td>
<td>Calculate the similarity between items and choose the items that are most similar to the target-user</td>
<td>Item recommendations</td>
<td>Retrieval of item ratings, and recommendation of items ranked</td>
<td></td>
</tr>
<tr>
<td>Hybrid-based recommendations</td>
<td>Both</td>
<td>Users' interests and item characteristics</td>
<td>Hybrid recommendations</td>
<td>Retrieval of both user and item ratings, and recommendation of hybrid ranked</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3. The specific recommendation process can be divided into 3 steps:

1. First, we need to obtain users' immediate preferences. We can forecast users' immediate ratings to items according to the position or other factors that we can get immediately, and reject some items that don't meet the reality, which is helpful to improve the efficiency of searching.
2. Second, it is very important to distinguish users' identities. In this step, we need judge whether the user is the registered member or not based on the theory that mobile phone number is the unique identification of our users.
3. Third, the last step is to do recommendations. At the beginning of this step, we take different ways to recommend for users according to their different identities. If you are a newer and have no history information, the system will sort the items on the basis of immediate preferences obtained in the first step, and choose the most possible interesting items as recommendation. If position is the measure of preferences, we choose those whose position rating is the highest, which is really effective to solve the cold-start problem. Else, if you are not a newer and have ever visited our sites before, we should get your historical ratings about items and related purchasing histories from users' history profile, and cluster integrating users' immediate interests, then search the clearest neighbor. After that, we should forecast users' ratings for items from their neighbors, and sort them by ratings, then select the former n ones. In the end, some of them that have been recommended to the user before should be removed to avoid repeated recommendation, the others are recommended to the user. Meanwhile, record the recommendations into users' historical profile.
III. THE BASIC ACO ALGORITHM

In this section we introduce the basic ACO algorithm. We decided to use the well-known traveling salesman problem as benchmark, in order to make the comparison with other heuristic approaches easier. Given a set of ri towns, the TSP can be stated as the problem of finding a minimal length closed tour that visits each town once. We call $d_{ij}$ the length of the path between towns i and j. In the case of Euclidean TSP, $d_{ij}$ is the Euclidean distance between i and j (i.e., $d_{ij} = \sqrt{(x_i-x_j)^2 + (y_i-y_j)^2}$). An instance of the TSP is given by a graph(N,E), where N is the set of towns and E is the set of edges between towns (a fully connected graph in the Euclidean TSP). Let $b_i(t)$ ($i=1,2,\ldots,n$) be the number of ants in town at time t and let $m = \sum_{i=1}^{n} b_i(t)$ be the total number of ants. Each ant is a simple agent with the following characteristics: it chooses the town to go to with a probability that is a function of the town distance and of the amount of trail present on the connecting edge to force the ant to make legal tours, transitions to already visited towns are disallowed until a tour is completed (this is controlled by a labu list). When it completes a tour, it lays a substance called trail on each edge (i,j) visited.

Let $\tau_{ij}(t)$ be the intensity of trail on edge (i,j) at time t. Each ant at time t chooses the next town, where it will be at time t+1. Therefore, if we call an iteration of the ACO algorithm the m moves carried out by the m ants in the interval (t,t+1), then every n iterations of the algorithm (which we call a cycle) each ant has completed a tour. At this point the trail intensity is updated according to the following formula:

$$\tau_{ij}(t+n) = \rho \tau_{ij}(t) + \Delta \tau_{ij}$$

(1)

where $\rho$ is evaporation is a coefficient such that $(1- \rho)$ represents the of trail between time t and t+n

$$\Delta \tau_{ij} = \sum_{k=1}^{m} \Delta \tau_{ij}^k$$

(2)

Where $\Delta \tau_{ij}^k$ is the quantity per unit of length of trail substance (pheromone in real ants) laid on edge (i,j) by the k-th ant between time t and t+n. It is given by

$$\Delta \tau_{ij}^k = \begin{cases} Q/L_k & \text{if } k\text{-th ant uses} \\ 0 & \text{edge } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \\ \end{cases} \text{ between time } t \text{ and } t+n$$

(3)
where $Q$ is a constant and $L_k$ is the tour length of the $k$-th ant. The coefficient $\rho$ must be set to a value $\rho < 1$ to avoid unlimited accumulation of trail. In our experiments, we set the intensity of trail at time 0, $\tau_{ij}(0)$, to a small positive constant $c$.

In order to satisfy the constraint that an ant visits all the $n$ different towns, we associate with each ant a data structure called the tabu list, that saves the towns already visited up to time $t$ and forbids the ant to visit them again before $n$ iterations (a tour) have been completed. When a tour is completed, the lobo list is used to compute the ant's current solution (i.e., the distance of the path followed by the ant).

The tabu list is then emptied and the ant is free again to choose. We define tabuk the dynamically growing vector, which contains the lobo list of the $k$th ant, tabuk the set obtained from the elements of tabuk and tabuk(s) the $s$-th element of the list (i.e., the $s$-th town visited by the $k$-th ant in the current tour).

We call visibility $\eta_{ij}$ the quantity $\eta_{ij}$. This quantity is not modified during the run of the ACO algorithm, as opposed to the trail, which instead changes according to the previous formula (1). We define the transition probability from town $i$ to town $j$ for the $k$-th ant as

$$p_{ij}^k(t) = \frac{\tau_{ij}^k(t) \cdot \eta_{ij}^k}{\sum_{j \in allowed_i} \tau_{ij}^k(t) \cdot \eta_{ij}^k}$$

where

$$allowed_i = \{N - tabu_k\}$$

And where $\alpha$ and $\beta$ are parameters that control the relative importance of trail versus visibility. Therefore the transition probability is a trade-off between visibility (which says that close towns should be chosen with high probability, thus implementing a greedy constructive heuristic) and trail intensity at time $t$ (that says that if on edge $(i, j)$ there has been a lot of traffic then it is highly desirable, thus implementing the autocatalytic process).

In traditional ACO Algorithm, the initialization of the pheromone matrix is equal. Ants need iterate many numbers to find the best tour. We can generate a large amount of tours (e.g. 100 tours), and then we choose some better tours (e.g. 30 tours). At last ants lay trail only on these better tours. These trails affect following ants.

When ant completes a tour, it always lays called trail on each edge visited. If the tour is worse, ant also lay the trail on each edge. These trails disturb the following ants, so the ACO algorithm's convergent speed is very slow. We can calculate the length of tour firstly, and then we compare with the given value. If the length of tour is less than the given value, we update trail values. Otherwise don't update the trail values.

The advantage of PSO algorithm method is that it use self information, individual best information and global best information. We can learn from genetic algorithm, and were rearrange formula (12). The $w \times v_d$ can be regarded as mutation operator, and $c_1 \times rand_1 \times (p_d - x_d) + c_2 \times rand_2 \times (p_d - x_d)$ can be regarded as crossover operator. But the crossover operator happened between the individual with local optimum and global optimum.

IV. EVALUATION OF PROTOTYPE AND ANALYSIS

A prototype system that recommends restaurants to customers based on the multi-agent architecture mentioned in section 3 is described in this section. The restaurant recommendation system provides modules of restaurant catalog presentation, keywords searching, user registration and login, mobile phone vouchers downloading, user rating and restaurants recommendation. These modules can help to collect comprehensive customer information to create generic user profile.

In our current prototype system, four recommendation algorithms are integrated to provide hybrid recommendation. They are location-based algorithm which could recommend restaurants according to user's location, collaborative recommendation algorithm that calculates user's similarity and makes restaurant recommendations according to users' experiences with similar preferences, content-based algorithm that recommends the restaurants according to the similarity of restaurants, and demographic recommendation algorithm which classifies users into groups according to their demographic information, such as sex, age, income level etc.

In the prototype system, the cooperating agents and their communication are implemented on the JADE-LEAP (Java Agent Development Environment and Lightweight Extensible Agent Platform). JADE-LEAP Agent containers are deployed to provide an agents supporting atmosphere. The prototype system stores many mobile device profiles with which the system user interface could be presented according to the characteristics of user's mobile devices.

In order to evaluate the performance of our restaurant recommendation system, 30 users and 150 restaurants are involved. In our system, users simulate their habits to go to restaurant. They can search the restaurant information by themselves or accept the recommendation given by the system. When they find their favorite restaurants, they can download corresponding vouchers to their mobile phones. And they can also rate some restaurants they have visited. The precision and recall are conflicting properties, high precision means low recall and vice versa. To find a trade-off between precision and recall, F-measure that is the weighted harmonic mean of precision and recall is used here.
From Fig. (4), we can see, the recommendation performance of our system improves a lot with time. It means the designed hybrid recommendation has learning ability through improving performance with user feedbacks. Moreover, F-measure does not change its go-up trend although there are new users entering in the system in every stage. This implies, in some extent, that new users do not influence the average system performance very much because they can use the experiences of existing users.

V. CONCLUSION

This paper proposed new personalized recommendation system architecture of mobile business application, and reach the conclusion that it is helpful to provide effective recommendations to users in mobile business by integrating it with the specific simulation example. In comparison with traditional E-Commerce architecture, its features can be concluded like below:

First, it's a good way to solve the cold-start problem. Every mobile phone or terminals has a definite user, whose personal profile can be stored in the mobile device possessing the unique identifier and users' online behaviors, which is the unique feature of mobile business. And it is really effective in resolving cold-start of personalized recommendation.

Second, it can optimize the problem of data sparsity to some extent. Users' ratings for items contain position ratings, which can be obtained in real-time. So, data sparsity can be improved by this. The next work is to build a general module of users, optimize the process of algorithm, and enhance the versatility of the architecture.

ACKNOWLEDGMENT

This work is supported by the Key Project of Guangxi Social Sciences, China (No.gxsk201424), the Education Science fund of the Education Department of Guangxi, China (No.2014JGA268), and Guangxi Office for Education Sciences Planning, China (No.2013C108).

REFERENCES