Computer Science and Information Systems 18(2):517–534 https://doi.org/10.2298/CSIS200119008T

Collaborative Filtering Recommendation Algorithm in Cloud Computing Environment

Pei Tian[1]

Human Resource Center, Xi'an Peihua University,
Xi'an, 710125, Shaanxi, China
tianpei@peihua.edu.cn

Abstract. With the advent of the era of cloud computing, the amount of application data increases dramatically, and personalized recommendation technology becomes more and more important. This paper mainly studies the collaborative filtering detection algorithm in the cloud computing environment. The algorithm migrates the collaborative filtering detection technology and applies it to the cloud computing environment. It shortens the recommendation time by using the advantages of clustering. A new recommendation algorithm can improve the accuracy of recommendation, and proposes a parallel collaborative filtering recommendation algorithm based on project. The algorithm is designed with programming model The experimental results show that the proposed algorithm has shorter running time and better scalability than the existing parallel algorithm.

Keywords: collaborative filtering, Hadoop, HDFS, Map Reduce, person correlation coefficient

1. Introduction

The rapid growth of Internet scale causes the problem of information overload [1]. It is difficult for users to obtain valuable information from massive data. As an important filtering method, personalized recommendation actively recommends articles of interest to users by analyzing their interests and historical behaviors. In order to solve the problem of Internet information interest rate overload [2], at present, different recommendation methods can be divided into There are three kinds of recommendation: content-based recommendation, collaborative filtering recommendation and hybrid recommendation [3, 4]. Recommendation based on collaborative filtering is undoubtedly the most successful personalized recommendation technology [4].

David Goldberg and others proposed a user collaborative filtering algorithm in [5]. The number of projects is far greater than the number of users, but for online e-commerce sites, the number of projects is often lower than the number of users. Location-based collaborative filtering recommendation can not only shorten the calculation time of recommendation, but also produce a variety of recommendations. Greg Linden et al. Proposed a Item-based Collaborative Filtering Recommendation Algorithm in [6]. By calculating the similarity between Items, we can find the similar

[1] Corresponding author

neighbors of Items and recommend them to active users. Because the number of items purchased by users only accounts for a small part of the total number of items, due to the sparsity of data, the recommendation neighborhood of collaborative filtering based on memory is not accurate. Sandvig j et al. Proposed a collaborative filtering recommendation algorithm based on Association Rule Mining in [7]. In the case of noise, the recommended accuracy shows better stability and robustness [8]. In order to improve the real-time performance of the recommendation algorithm, document [9] implements the Item-based Collaborative Filtering Recommendation Algorithm in parallel on the Hadoop platform. However, in the algorithm designed in reference [9], the calculation of all users' purchase item pairs is to find all users who purchase the item pair by searching all purchase records of the two items. The time complexity of this method is O ($m^2 * n^2$) (M represents the number of users, n represents the number of items). At the same time, the algorithm uses two Map Reduce processes to calculate the prediction score of the user's non purchased items by regarding the user as the center: sending the items purchased by the same user to the same node, reading the similarity matrix into memory and the similarity of similar items into each item of the user when calculating the prediction, which not only reads too much redundant data , and in the massive data environment, the number of users and goods are very large, the scale of similarity matrix will lead to it can not read all, will affect the scalability of parallel collaborative filtering algorithm [9].

Based on the above problems, this paper proposes a Parallel Collaborative Filtering Recommendation Algorithm based on Item (Parallel-Item-Base Collaborative Filtering, PIB-CF), which will use Map Reduce framework and HDFS to implement the Item-based Collaborative Filtering Algorithm in Hadoop cluster. In view of the problem of algorithm design in reference [10], PIB-CF algorithm calculates that the user purchases the item pair together by sending any two item pairs as keys to the reduce node, and the items with the same item pair will be sent to the same node. The time complexity of this method is O ($n^2 * m^2$), which will greatly reduce the search time. And this paper designs an efficient parallel strategy, designs a Map Reduce process to achieve the prediction and scoring function of two Map Reduce processes in reference [10]. In this way, the calculation results stay in memory, thus reducing the traffic, improving the efficiency of the algorithm, and achieving good real-time and scalability. Experimental results show that PIB-CF algorithm is effective.

In the first part of this paper, collaborative filtering recommendation technology and Hadoop cloud computing platform are introduced. In the second part, Project-based Collaborative filtering recommendation algorithm is proposed. In the third part, experiments are carried out. In the fourth part, the algorithm proposed in this paper is summarized.

2. Related Research

2.1. Collaborative Filtering Recommendation Technology

The idea of collaborative filtering technology is simple and easy to understand, and it is recommended for individuals from the perspective of groups [11]. The basic idea is that

if users had the same preferences in the past, they would have more similar preferences in the future. For example, if there are users a and B, their purchase experience is very similar. When user a recently obtains a book, but user B does not, then the recommendation idea is to recommend this book to B. Recommend books that you may like, which requires filtering out the most likely books from a large number of collections. Users are collaborating with others implicitly, so this technology is also called CF, collaborative filtering. Collaborative filtering algorithms can be divided into three categories: memory based recommendation algorithm, model based recommendation algorithm and hybrid recommendation algorithm, as shown in Figure 1.

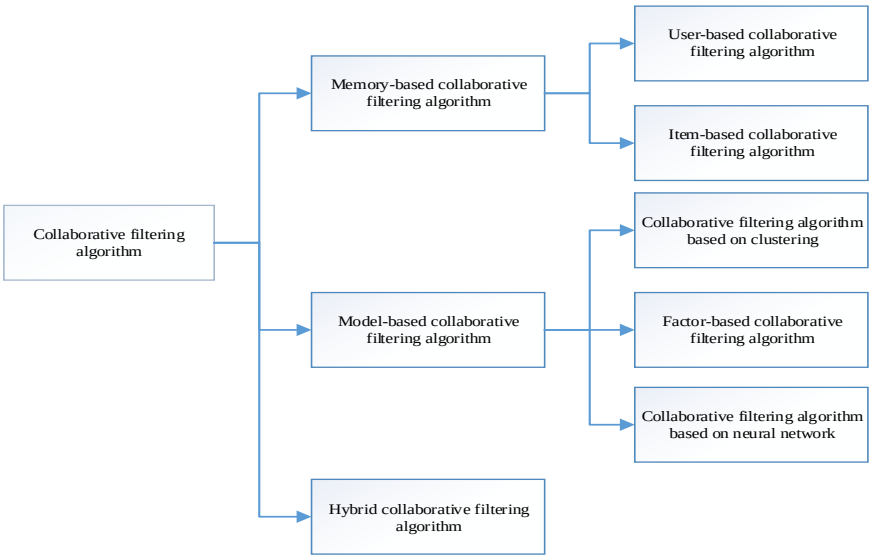

Fig. 1. Classification of collaborative filtering algorithm

Memory based collaborative filtering algorithm

Memory based collaborative filtering algorithm is a successful and practical recommendation algorithm. It is recommended by collaborative filtering through the saved historical information in the system. It analyzes and calculates the relevant historical information of the Item and users, generates a similar set of users or Item neighbors, and calculates and generates recommendations according to the scoring situation. In this process, there are two algorithms: user based recommendation algorithm (User-CF) and item based recommendation algorithm (Item-CF). The two algorithms have the same calculation flow, as shown in Figure 2:

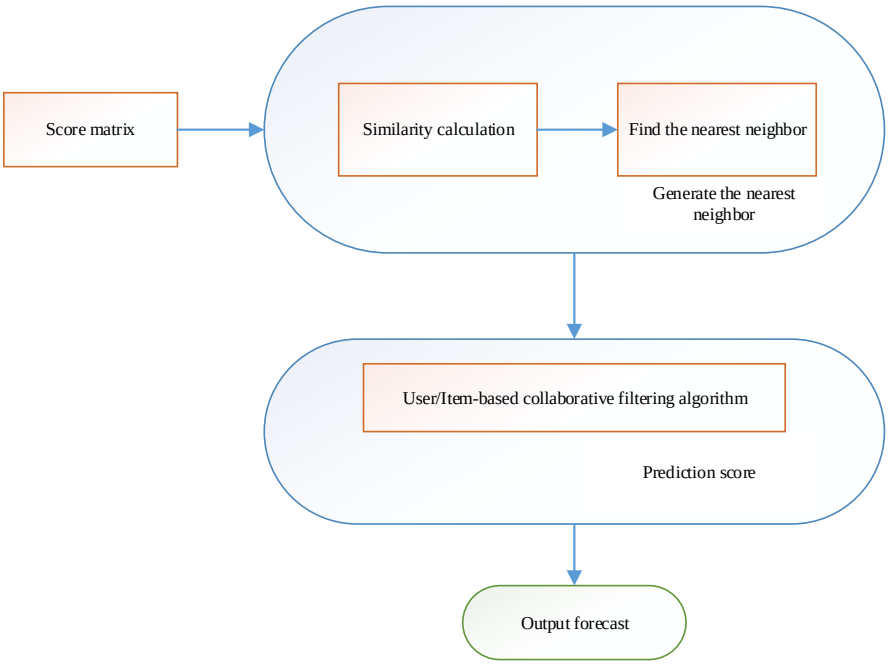

Fig. 2. Memory based collaborative filtering algorithm

User based recommendation algorithm

In real life, a person with needs often consults other people who have the same needs, and makes personal decisions by referring to other people's opinions and methods. In the same way, when a group of people with the same interests gather together and recommend their own things, it is likely that this is what other people who do not have this item need. In fact, the user-based recommendation algorithm follows this idea: if there are some users, their scoring items are basically the same, which indicates that they have the same interest preference, then their unique scoring items can be used as other users' Item scoring reference. The collaborative filtering algorithm based on users is from the perspective of users, which refers to users' ratings of items similar to the interests of the target users, to calculate the user's rating of the non rated item, namely Formula 1.

$$\hat{r} = \bar{r} + \frac{\sum_{v \epsilon S(u,K) \cap N(i)} W_{uv}(r_{vi} - \overline{r_v})}{\sum_{v \epsilon S(u,K) \cap N(i)} |W_{uv}|} \quad (1)$$

Where, *S (u, K)* is the set of K users whose interests are most similar to those of u, and K needs to be defined according to the data set. *N (i)* is a collection of users who overestimate Item I. r_{vi} is the rating of user V on Item i. $\overline{r_v}$ is the mean value of all items of user v. W_{uv} represents the similarity between user u and user v. generally.Generally, we choose one of Euclidean distance, cosine similarity measure and Pearson correlation coefficient as similarity calculation method.

Item based recommendation algorithm

Item-based recommendation is based on the degree of association between projects, which is different from user recommendation based on the similarity between users. For example: when a user likes item a, and others like item a also like Item B, then recommend Item B to the past. Simply put, "what Item is similar to what user a likes". Figure 3 shows the difference between Item-based and user-based recommendation algorithms: find similar users and know what they prefer based on user's recommendation (solid line); know user's preference based on Item recommendation (dotted line), and find similar Items.

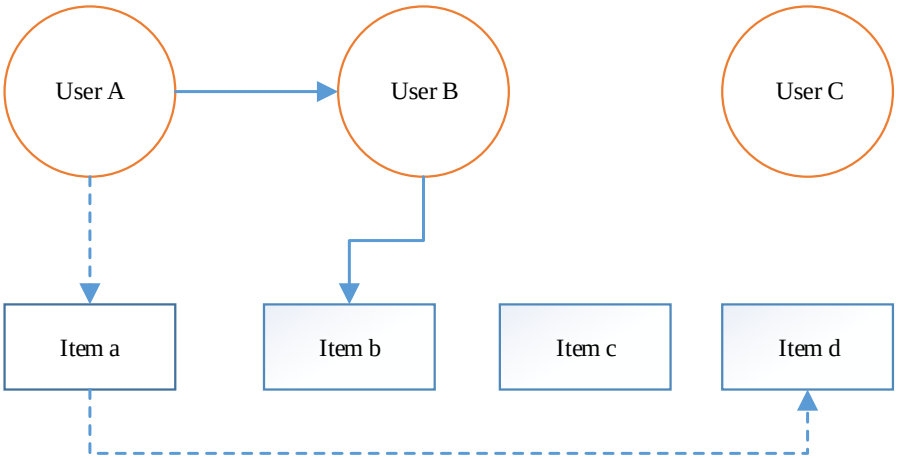

Fig. 3. Difference between item based and user based

The Item-based recommendation algorithm is to recommend from the perspective of the item. Like user based recommendation algorithm, one of Euclidean distance, cosine similarity and Pearson coefficient is used as the similarity parameter between items. Then calculate the score of the user who has not scored the item, that is, formula 2.

$$\hat{r}_b = \bar{r}_b + \frac{\sum_{a\epsilon I\,(a,K)\cap M\,(i)} W_{ba}\,(r_{ai} - \overline{r_a})}{\sum_{a\epsilon I\,(a,K)\cap M\,(i)} |W_{ba}|} \quad (2)$$

This formula is the score of the User I of forecast item b. $I(a, K)$ is the neighbor set of Item a, which has k neighbors. $M(i)$ is a collection of items that are overrated by user I. r_{ai} is the rating of user a on Item I. $\overline{r_a}$ is the mean value of all user scores of item a. W_{ba} represents the similarity parameter between items b and a.

2.2. Similarity Algorithm Research

For similarity algorithm, there are Euclidean distance, cosine similarity measurement, Pearson correlation coefficient similarity algorithm and so on. At present, Pearson coefficient similarity algorithm is widely recognized and applied. Next, we will introduce each algorithm and analyze its advantages and disadvantages.

European distance

This similarity method is a kind of calculation method which is widely used in many fields. It mainly uses users or projects as points on the plane to calculate the distance between each point. The calculation method is shown in Formula 3.

$$sim(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_m - b_m)^2} \quad (3)$$

Where, $a, b \in U$ and a_i represent user a's rating for item i. The value range of sim(a, b) varies greatly and may be greater than 1, which is not conducive to the calculation of prediction score. Therefore, the sweight value calculated by formula 4 will be used as the weight value in the prediction calculation, so that the weight value is within [0,1].

$$sweight = \frac{1}{1+sim} \quad (4)$$

Through Euclidean distance, although we can get the similar neighbor set, it has no too high credibility as the weight of the prediction calculation, which will seriously affect the accuracy of the prediction score.

Cosine similarity

The measure of similarity is to calculate the cosine of the angle between two n-dimensional vectors. When the angle between vectors is smaller, the cosine value will be closer to 1, and the directions of the two vectors are more similar. For example, if there are two users a and B, their corresponding scoring vectors are $\vec{a} = \{a_1, a_2, \ldots\ldots\}$ and $\vec{b} = \{b_1, b_2, \ldots\}$,the similarity can be defined as formula 5.

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|} \quad (5)$$

The sign · represents the dot product between vectors, and $|\vec{a}|$ represents the Euclidean length of vectors, that is, the square root of the dot product of vectors themselves. The value of cosine similarity is between 0 and 1. The closer one is, the more similar it is. The accuracy of the prediction score is relatively high, but because the method does not take into account the difference between the average user score, there are still some defects. For example, in the actual score, the five values of {1, 2, 3, 4, 5} are taken as the substitute value of {worst, poor, general, good, best}. Some people are used to playing {1, 2, 3} such {bad, general, good} score, which will have a certain impact on the score prediction calculation.

Pearson correlation coefficient [15]

Pearson correlation coefficient can be used as a parameter to measure the linear correlation between two variables, such as formula 6.

$$W_{a,b} = \frac{\sum_{i \in P} (r_{a,i} - \overline{r_a}) \cdot (r_{b,i} - \overline{r_b})}{\sqrt{\sum_{i \in P} (r_{a,i} - \overline{r_a})^2 \cdot \sum_{i \in P} (r_{b,i} - \overline{r_b})^2}} \quad (6)$$

Where the $\overline{r_a}$ symbol represents the average score of user a. The range of Pearson correlation coefficient is [-1,1]. When its absolute value tends to 1, it shows that neighbors are more similar. For this method, it takes into account the fact that the user

rating standards are not the same. It can find obvious user relationships, and then get better and accurate similarity.

Compared with the other two methods, Pearson correlation coefficient is more consistent with the recommended algorithm, and it is also the most commonly used in collaborative filtering.

2.3. Hadoop Cloud Computing Platform

Cloud computing is a pay as you go model (IT resources include networks, servers, storage, applications and services) that provide easy access to a customizable pool of IT resources through the Internet. These resources can be deployed quickly and require little management or interaction with service providers. Cloud computing centralizes the distributed computing, storage, service components and network software resources on the network. Based on resource virtualization, cloud computing provides convenient and fast services for users. It can realize the distributed and parallel processing of computing and storage. If "cloud" is regarded as a virtualized storage and computing resource pool, then cloud computing is the data storage and network computing services provided by this resource pool for users based on network platform. The Internet is the largest "cloud", on which a variety of computer resources constitute a number of huge data centers and computing centers.

The framework of Hadoop [12] includes HDFS, Map Reduce, H Base and other open-source components. At the same time, Hadoop also includes several independent subsystems, such as: hive, Chukwa, Avro, etc. So many components and subsystems make up a powerful cloud computing platform [13], Hadoop, as shown in Figure 4.

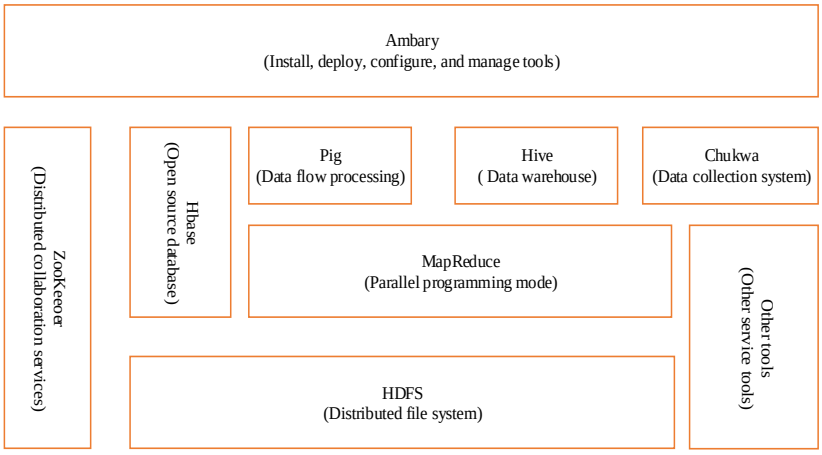

Fig. 4. Hadoop platform structure

HDFS distributed file system

HDFS is the core sub Item of Hadoop Item. HDFS is born out of Google File System (GFS), which reduces the demand for hardware. It only needs cheap hardware facilities,

and can provide a distributed file system that can store and process massive data through network link [14]. HDFS is built with master / slave structure, as shown in Figure 5.

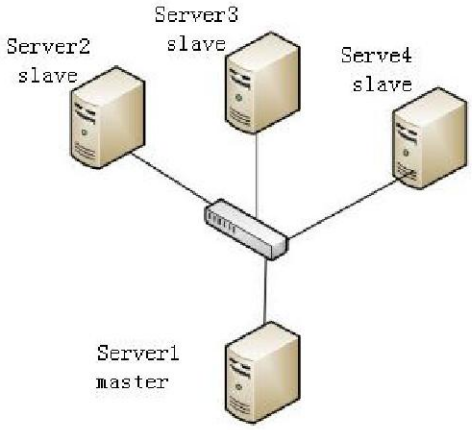

Fig. 5. Master-slave structure

For general machine deployment [16], there is only one primary node named node on the master, and each slave, there is a slave Data node on. A complete HDFS cluster consists of a name node and several data nodes. Name node is a main control node, which is responsible for the namespace of HDFS, and also manages the file tree of HDFS, metadata of all files and directories, which acts as an administrator of HDFS system. Data node is the node of data storage in file system [17]. For the storage of large-scale data files, HDFS will first split the whole data file to get multiple data blocks, and then complete redundant backup storage in multiple Data nodes. HDFS clients can complete the read and write operations of data files in the system by calling the API provided by the system. First, the client accesses the node Name node, reads all the metadata information saved, finds the corresponding Data node, and reads and writes the data. The architecture is shown in Figure 6.

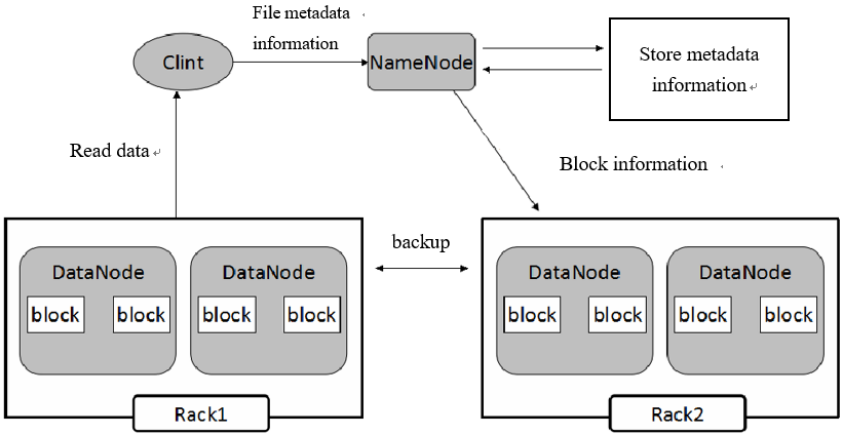

Fig. 6. Structure of HDFS

In addition, HDFS has several other functions in data security management:

(1) Data backup mode: file block is the storage form of HDFS data. Generally, there are three backups for a file block, which can greatly improve the safety factor of data;

(2) Heartbeat detection: detect Data Node frequently and in real time. If there is a problem with Data Node, use data backup to ensure the integrity of data;

(3) Data verification: CRC32 method is used for data verification. For file blocks, not only the data but also the corresponding verification information will be written; when reading file blocks, the verification will be performed first and then read.

(4) Security mode: the system in security mode will be restricted in permissions. Users are not allowed to modify and delete the contents of the system until the end of security mode. This is to detect the data on each Data Node and determine the validity of the data when the system starts.

Map reduce programming model

MapReduce is a cluster based high performance parallel computing platform (cluster infrastructure). It allows the use of common commercial servers on the market to form a distributed and parallel computing cluster with dozens, hundreds to thousands of nodes. It provides a huge but well-designed parallel computing software framework, which can automatically complete the parallel processing of computing tasks, automatically divide computing data and computing tasks, automatically allocate and execute tasks on cluster nodes and collect calculation results, and hand over many complex details of the underlying system involved in parallel computing such as data storage, data communication, fault-tolerant processing and so on The system is responsible for processing, which greatly reduces the burden of software developers.

In Hadoop, there are two carriers for executing Map Reduce tasks: Job tracker and Task tracker. Job tracker is responsible for the supervision and control of Map Reduce. According to the requirements of Map Reduce, the corresponding resources are scheduled. In a Hadoop cluster, you cannot have two Job tracker tasks at the same time.

Task tracker is responsible for running Map Reduce and responding to the request of Job tracker. Here, Map Reduce is divided into two parts: Map task and reduce task. As shown in Figure 7, the control flow of Map Reduce is as follows: Job tracker schedules tasks to Task tracker, and when Task tracker performs tasks, it will return progress reports. Job tracker will record the progress status. If a task on a Task tracker fails to execute, Job tracker will assign the task to another Task tracker until the task ends.

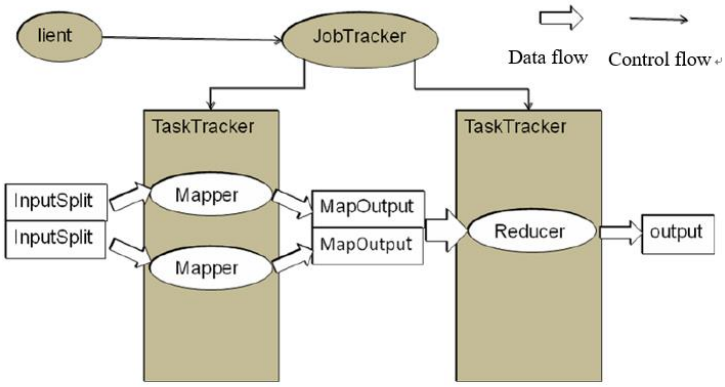

Fig. 7. Simple working principle of Map Reduce

The data processing flow of Map Reduce is as follows: the data is first processed into multiple Input Splits, and then input into the corresponding number of maps. A map program will read the data at the specified position of Input Split, then process the data in the set way, and finally write it to the local disk. Reduce will read the map output data, merge the data, and then output them to HDFS.

In Hadoop, each Map Reduce task is initialized as a job. Each job can be divided into two phases: the mapping phase and the restore phase. These two stages are represented by two functions, the Map function and the Reduce function. Map function receives input in the form of < key, value > and produces the same intermediate output in the form of < key, value >. Hadoop is responsible for passing all values with the same intermediate key value to the reduce function. The reduce function receives the input in the form of < key, (list of values) > and then processes the value set and outputs the result. The output of reduce is also in the form of < key, value >. In order to display conveniently, mark three < key, value > as $< k_1, v_1 >$, $< k_2, v_2 >$, $< k_3, v_3 >$, as shown in Figure 8:

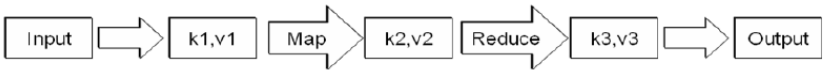

Fig. 8. Map Reduce programming model

3. Design and implementation of project based parallel collaborative filtering recommendation algorithm

3.1. Item based collaborative filtering recommendation algorithm

Item based collaborative filtering refers to matching the products purchased or rated by a user to similar products, and then entering similar products into the recommendation list. From the point of view of calculation, it is to take the evaluation of all users on the specified purchased goods as a vector, and each user's evaluation on the goods as the corresponding component in the vector. In this way, the similarity between items is calculated by the vector value, and the similar items of items are obtained. Then, the current user's evaluation on the non purchased items is predicted according to the user's historical shopping records, and a List of items sorted by forecast as recommendations. The algorithm can be divided into four parts:

1. Score data preprocessing, calculate the average score of the Item.
2. Through a certain similarity measurement method, the similarity between items is calculated.
3. Through a certain forecast scoring model, the forecast scoring of user's UN scored items is calculated.
4. Time performance measurement and precision measurement of the algorithm.

3.2. Parallelization of Collaborative Filtering Recommendation Algorithm Based On Item

The most time-consuming stage of collaborative filtering recommendation algorithm is to calculate similarity matrix and predict user preferences. If the number of users is m, the time complexity of similarity calculation is O (n^2m), and the time complexity of prediction is O (nm). Obviously, when calculating the similarity matrix, the similarity of one item pair is independent of that of another pair. So the similarity matrix can be calculated in parallel. Secondly, when predicting user preferences, computing preferences for different users are also independent of each other, so predicting user preferences can also be done in parallel. However, the calculation of similarity must be completed before the prediction of preference, so the two must be executed serially.

Map Reduce is a popular parallel programming model at present. Users only need to specify the calculation process of Map and function Reduce, and the system will automatically calculate in parallel on a large-scale cluster. And both implement Map Reduce.

Using Map Reduce model to implement PIB-CF, a parallel algorithm based on collaborative filtering recommendation, requires three map and reduce processes. Map1 calculates each user's purchase item and its score. Reduce1 calculates the mean value of item ID scores. Map2 is used to calculate all pairs of items in the same user's purchase record, and reduce2 is used to calculate the similarity between items. Map3 is used to calculate all other items similar to an item, and reduce3 is used to calculate the user's forecast score for items not scored. Map1 and reduce1 calculate the Item and its mean

value and save them in the file av-file. Because the calculation process is relatively simple, we will not go into details here.

The user file of user score data read by Map2 function is stored in. Each line of data represents a user's historical purchase item record. It is read into the map node line by line in the form of key value pair < key, value >. Each key value pair represents a data record. key is the offset of the current record relative to the starting point of the input data file, and value is the item ID and its score in the current record. Map2 < key, value > calculates all pairwise item pairs purchased by the current user and their corresponding scores. The pseudo code of the function is as follows:

```
Input: key, the value corresponding to the key;
Output: key, the value corresponding to the key;
The pseudo code flow is as follows:
Map2(key,value){
1. GenItemsandRatings(value ,item,rating,len);
2. For i=0 to len - 1 do
3.   For j=i+1 to len do {
4.       item_a=Items[i]; rating_a=Ratings[i];
5.          item_b=Items[j]; rating_b=Ratings[j];
6.          key'=CombineItems(item_a,item_b);
7.       Value'=CombineItems(rating_a,rating_b);
8.          Output<key,value>;
9.   }
10.}
```

After Map2 is output, the system will send item pairs with the same key to the same Reduce node. The function of Reduce (key, value) is to calculate the similarity between two corresponding items. The pseudo code of the function is as follows:

```
Input: item label key, item corresponding score value;
Output: similarity of item to key '
The pseudo code flow is as follows:
Reduce2(key,value){
1. AssignValue(value,rating_i,rating_j);
2. ReadFileAveRat(av-file,items,aveRatings);
3. for each val in value{
4.   ReadRatings(val,rating_i[k],rating_j[k]);
5.         k++;
6.         }
7.       value‘=Pearson();
8.  key'=key;
9.  Output<key,value>
10.}
```

After the implementation of Reduce2, the system will get the Item similarity matrix, which will predict and grade the items that are not evaluated by each user.

The output results of Reduce2 take the form of < key, value > as the input of map3. Key is the Item pair, and value is the similarity of the Item pair. The function of Map3 (key, value) is to send all Item pairs containing a Item to the same reduce node. The pseudo code of the function is as follows:

```
Input: item code key;
Output: Code of similar item corresponding to item key
'and its similarity list value';
Map3(key,value){
1. ReadItem(key,item1,item2);
2. key'=item1;
3. value'=Combine(item2,value);
4. Output<key',value'>;
5. Key'=item2;
6. Value'=Combine(item1,value);
7. Output<key,value>
}
```

After Map3 is executed, the system will send the items similar to the items and the similarity between them to the same Reduce node. Reduce3 (key, value) calculates the predicted score of the user's non scored items. The pseudo code of the function is as follows:

```
Input: item code key, similar item list value, user
rating file av-file, item average rating file av-file
Output: Item forecast score;
The pseudo code flow is as follows:
Reduce3(key,value){
1. ReadFileAveRat(av-file,allItems,aveRating);
2. ReadFileRat(user-file,userid,items,rating);
3. for each val in value{
4.   ReadItemSimi(val,item[k],simi[k]);
5.   k++;
6. }
7. if array items no item key {
8.    value‘=Prediction();
9.    key'=Combine(userid,key);
10. }
11. Output<Key',value'>
12.}
```

4. Experimentand analysis

4.1. Experimental platform and data set

In this paper, we use the dataset provided by the Movielens website (HTTP / /: www.grouplens. org / node / 73 /), which contains 71567 users and their evaluation of films. Each user evaluates at least 20 films with a rating of 1-5. The sparse degree of movie score in this dataset is 0.987. In this experiment, 21 (8-core) CPU servers constitute the parallel computing environment, and Hadoop is used to implement the parallel collaborative filtering recommendation algorithm proposed in this paper.

User rating data in the recommendation system includes user set U = {u_1, u_2 , …,u_m} and item set I = {i_1, i_2 , i_n}, the scoring matrix can be expressed as a m * n-order user scoring matrix M = $(r_{ij})_{m\times n}$, as shown in Table 1

Table 1. user Item rating matrix M

	I_1	…	I_j	…	I_n
U_1	$r_{1,1}$	…	$r_{1,j}$		$r_{1,n}$
…					
U_i	$r_{i,1}$	…	$r_{i,j}$		$r_{i,n}$
…					
U_m	$r_{m,1}$	…	$r_{m,j}$		$r_{m,n}$

Each element of the matrix represents the user's rating of the Item, and the rating reflects the user's preference for the Item. The score can be used to indicate whether you like it or not, or the number can be used to indicate how much you like it.

4.2. Measurement method

If u is used to represent the user set of item and common scoring, then the scoring similarity sim (i, j) of item i and j can be calculated by person correlation coefficient [11], and the formula is as follows:

$$sim(i,j) = \frac{\sum_{u\in U}(R_{u,i}-\overline{R_i})(R_{u,j}-R_j)}{\sqrt{\sum_{u\in U}(R_{u,i}-\overline{R_i})^2}\sqrt{\sum_{u\in U}(R_{u,j}-\overline{R_j})^2}} \quad (7)$$

Where r_{ui} and r_{uj} represent the user's u-scores for items i and j, respectively, $\overline{R_i}$, $\overline{R_j}$ represents the average scores for items i and j.

In order to make the prediction more accurate, this paper uses an optimized prediction scoring strategy: take the average score of the items that the target user needs to score as the benchmark value, then find the neighbor set of the target items of the target user, and then use the item similarity in the neighbor set as the weight value to calculate the prediction score of the target items, and predict the calculation of the evaluation value of the user u for the item I The following expressions are as follows:

$$P_{u,i} = \overline{R_i} + \frac{\sum_{j\in S(i)} sim(i,j)*(R_{u,j}-\overline{R_j})}{\sum_{j\in S(i)}|sim(i,j)|} \quad (8)$$

Where r_{ui} represents the user U's score for item I, $\overline{R_j}$ represents the average score of item j, and sim (i, j) represents the similarity between item i and item j.

4.3. Experimental results and analysis

In order to test the scalability of the system, two experiments are designed. In the first group of experiments, the number of computing nodes was set, and the size of test data set was variable. The second experiment set the size of data set, and the number of computing nodes was variable. Because the process of calculating similarity in collaborative filtering recommendation based on Item can be calculated offline, only the predicted user score needs to be calculated in real time, so the test time of this experiment is the time of predicting Item score. The experimental results are shown in Figures 9 and10.

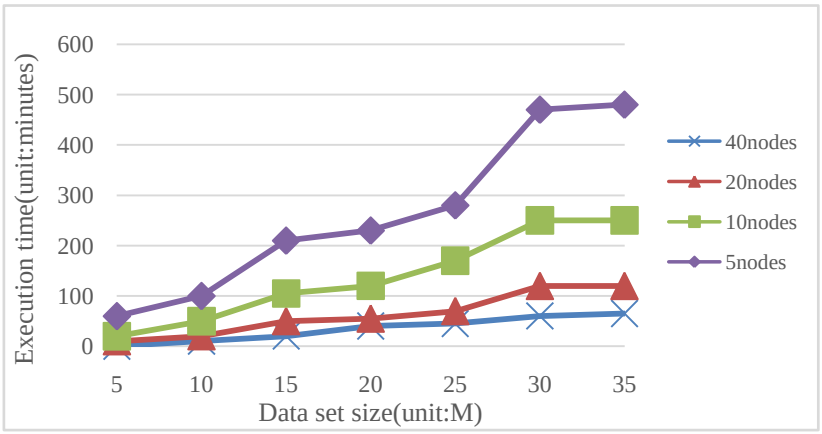

Fig. 9. Data volume and execution time

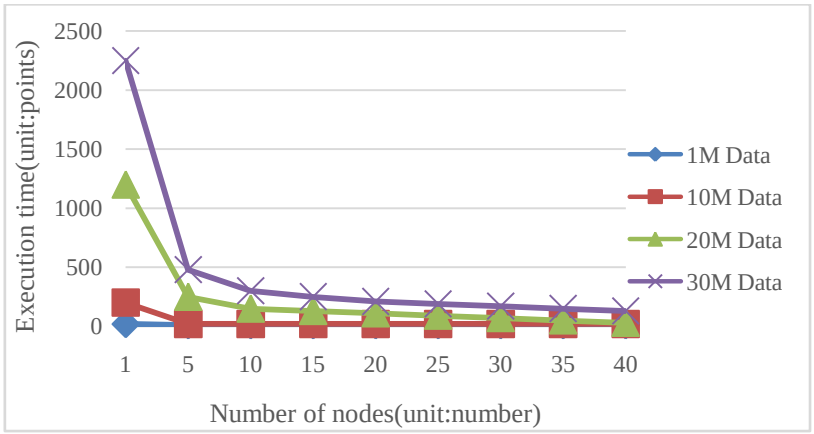

Fig. 10. Relationship between number of physical nodes and execution time

It can be seen from the figure and figure that the online prediction time can be greatly reduced after the collaborative filtering recommendation algorithm based on the Item is implemented in parallel, and good scalability can be achieved by increasing the number of computing nodes with the increasing amount of data.

In order to compare the time performance of the algorithm with the algorithm proposed in the literature, we designed a set of experiments. The experimental data set is 10 m, running on 1, 5, 10, 15, 20, 25, 30 nodes respectively. The comparison of algorithm execution time is shown in Figure 11, and the experimental results show that the performance of this algorithm is better than that of the reference [8].

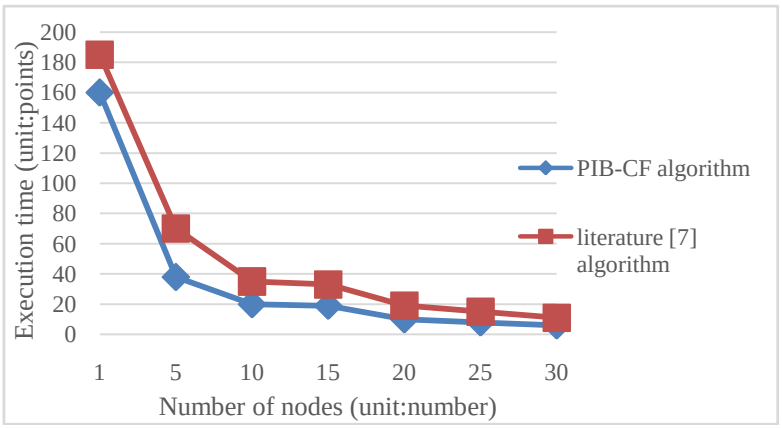

Fig. 11. Execution time comparison between PIB-CF algorithm and literature

In this chapter, a new parallel recommendation algorithm PIB-CF is proposed, which is implemented in Hadoop cluster. It can reduce the data communication and execution time by reasonably allocating the computation intensive similarity calculation process and Item prediction scoring process to each processing node. Experimental results show that PIB-CF algorithm is effective.

5. Summary

Cloud computing has become a hot technology in academia and industry. Many universities, scientific research institutions and Internet giants have invested in the research of cloud computing, which promotes the continuous upgrading from computer hardware to software, and promotes the development of computer and network technology. By purchasing cloud services, governments, universities and small enterprises do not need to purchase hardware resources, but only need a small amount of investment to purchase services from cloud service providers. This can not only reduce capital investment and make full use of existing resources, but also obtain complete solutions from cloud service providers. With enterprises turning to cloud platforms, users can enjoy the convenience of cloud computing and cloud storage anytime and anywhere through mobile terminals. The emergence of cloud computing technology has become an effective solution to solve the problem of big data. Combine cloud computing technology, recommendation technology and data mining technology.

Through the method of data mining, using the huge computing and storage advantages of cloud computing, the recommendation system can get faster and more accurate recommendation results. In this paper, a parallel collaborative filtering recommendation algorithm PIB-CF is proposed. This paper parallelizes the Item-based Collaborative Filtering Algorithm on Hadoop platform. The existing parallelization method is improved, which makes the improved algorithm run faster and more practical. Moreover, the experimental results show that the improved algorithm has good scalability.

Through cloud computing technology, the collaborative filtering algorithm based on project is parallelized in the platform. And through the hybrid recommendation technology, the recommendation accuracy is effectively improved. However, due to the limited research time, there are still some problems to be studied: the sparsity of item rating matrix is the biggest problem faced by the recommendation system. Because the items that users have purchased or browsed occupy only a small part of the total project, the similarity between users or projects cannot reflect the real relationship between them correctly. Therefore, matrix filling becomes the key problem to improve the accuracy of recommendation. The problem of cold start can not be solved effectively in the recommendation system. For a new user, due to the lack of historical information, how to obtain high-precision and comprehensive recommendation from less historical information has become a hot issue in the research of recommendation system. In this paper, we propose a hybrid recommendation algorithm to improve the recommendation accuracy. Moreover, the cold start problem is considered at the beginning of the algorithm design. However, due to the lack of research time, it was not able to complete. In the future research, we will continue to complete the research of cold start.

References

1. Lv, Zhihan, Weijia Kong, Xin Zhang, Dingde Jiang, Haibin Lv, and Xiaohui Lu.:Intelligent Security Planning for Regional Distributed Energy Internet. IEEE Transactions on Industrial Informatics, (2019)
2. Tunkelang D.: Recommendations as a Conversation with the User//Proc of the 5th ACM Conf on Recommender Systems. New York: ACM, 11-12. (2011)
3. Adomavicus G, Tuzhilin A.:Tpward the Next Generaton Od Recommender Systems: A Survey of the State-of-the-art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734-749.(2005)
4. Balabanović M, Shoham Y.: Fab: Content-Based, Collaborative Recommendation. Communications of the ACM, 40(3), 734-749.(1997)
5. Goldberg D, Nichols D, Oki B M.: Using Collaborative Filtering to Weave an Information Tapestry. Communications of the Acm, 35(12), 61-70.(1992)
6. Linden G, Smith B, York J.: Amazon. Com Recommendations: Item-to-item Collaborative Filtering. Internet Computing, IEEE, 7(1), 76-80.(2003)
7. Sandvig J J, Mobasher B, Burke R.: Robustness of the 2007 ACM Conference on Recommender Systems. Minneapolis, USA. Acm,105-112.(2007)
8. Wu, Y., Rong, B., Salehian, K., & Gagnon, G.: Cloud Transmission: A New Spectrum-Reuse Friendly Digital Terrestrial Broadcasting Transmission System. IEEE Transactions on Broadcasting, 58(3), 329-337. (2012)
9. Zhang, Y., He, Q., Xiang, Y., Zhang, L. Y., Liu, B., Chen, J., &Xie, Y.: Low-cost and Confidentiality-Preserving Data Acquisition for Internet of Multimedia Things. IEEE Internet of Things Journal, 5(5), 3442-3451. (2017)

10. Jiang J, LU J, Zhang G,.: Scaling-up Item-based Collaborative Filtering Recommendation Algorithm Based on Hadoop. Proceedings of 2011 IEEE World Congress on Services. Washington Marriott, DC, USA. IEEE, 490-497.(2011)
11. Resnick, Sushak.:GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 Computer Supported Collaborative Conference.(1994)
12. Lu Jiaheng.: Hadoop practice. Beijing: China Machine Press.(2012)
13. Han, B., Li, J., Su, J., & Cao, J.: Self-supported Cooperative Networking for Emergency Services in Multi-hop Wireless Networks. IEEE Journal on Selected Areas in Communications, 30(2), 450-457.(2012)
14. Konstantin S, Kuang H, Radia S.: The HadoopDistributed File System. Symposium on Massive Storage SystemsandThechnologies, 23(7), 385-394.(2010)
15. Ahn H J.: A New Similarity Measure for Collaborative Filtering to Alleviate the New User Cold-starting Problem. Information Sciences,178(1), 37-51.(2008)
16. Yue, L.: Cable-driven MIS Robot with Intuitive Manipulability and Direct Force Feedback. Investigación Clínica, 60(4),813-824.(2019)
17. Han, Biao, Jie Li, Jinshu Su, Minyi Guo, and Baokang Zhao.: Secrecy Capacity Optimization via Cooperative Relaying and Jamming for WANETs.IEEE Transactions on Parallel and Distributed Systems, 26, No. 4, 1117-1128.(2014)

Pei Tian was born in Xi'an, Shaanxi. P.R. China, in 1983. He received the master's degree from Northwest University of political science and law, P.R. China. Now, he works in the human resources center, Xi'an Peihua University. His research interest include cloud computing and e-commerce collaborative filtering. E-mail: tianpei@peihua.edu.com

Received: January 19, 2020; Accepted: December 20, 2020