# Personalized Recommendation System on Hadoop and HBase

Shufen Zhang(✉), Yanyan Dong, Xuebin Chen, and Shi Wang

College of Science, North China University of Science and Technology, Tangshan, Hebei, China
{hblgzhsf,453705197,chxb}@qq.com, ws10121@126.com

**Abstract.** In view of the existing recommendation system in the Big Data have two insufficiencies: poor scalability of the data storage and poor expansibility of the recommendation algorithm, research and analysis the IBCF algorithm and the working principle of Hadoop and HBase platform, a scheme for optimizing the design of personalized recommendation system based on Hadoop and HBase platform is proposed. The experimental results show that, using the HBase database can effectively solve the problem of mass data storage, using the MapReduce programming model of Hadoop platform parallel processing recommendation problem, can significantly improve the efficiency of the algorithm, so as to further improve the performance of personalized recommendation system.

**Keywords:** Hadoop · HBase · MapReduce · Personalized recommendation

## 1 Introduction

With the innovation of Internet technology, information resources on the network has become more and more abundant, it makes people step into the ocean of data - the Era of Big Data. In the era of big data, it becomes extremely difficult for people to find the information they need from the vast amounts of data quickly and efficiently. This problem is also known as Information Overload. Personalized recommendation technology using the existing historical data and real-time data, analysis of each user's behavior intention, and establish a corresponding interest model for each user. According to users' interest model push potential items, realize the personalized recommendations.

The traditional personalized recommendation system has two insufficiencies: poor scalability of data storage and poor expansibility of recommendation algorithm. This's because traditional personalized recommendation system underlying data storage using a Relational Database Management System, aka RDBMS. Over the system time running, the number of users and projects in the system will continue increasing; eventually leading RDBMS systems is difficult to load mass data storage. Moreover, the execution speed of recommendation algorithm is the core part of the personalized recommendation system; it directly affects the accuracy and efficiency of the recommendation. However, traditional recommendation system using a single node serial operation, it is hard to meet user demand for real-time recommendation result in the huge amounts of data environment.

DOI: 10.1007/978-981-10-0457-5_5

In view of two aspects issue above-described in the traditional personalized recommendation systems, this paper proposes a personalized recommendation system based on Hadoop and HBase. The underlying system used HBase database manages mass and sparse user behavior data, use of HBase column oriented storage and storage scalability characteristics to solve the poor scalability problems that exist in the traditional recommendation system. Data processing using Hadoop distributed computing advantage change every stage of the recommendation algorithm to parallel processing. Improve the execution efficiency of algorithm, reduce recommended time consuming process, enhance scalability of data processing, make recommendation system can better adapt to the practical application of huge amounts of data.

## 2 Personalized Recommendation System and Hadoop, HBase

### 2.1 Personalized Recommendation System

In recent years, with the constant innovation of new network technology, personalized recommendation system is becoming more and more widely used in our daily life, resulting in personalized recommendation technology become a challenge and opportunity coexist academic research field. A complete personalized recommendation system should include three parts: system show page, algorithm calculation engine, and data storage module. Recommendation system architecture diagram is shown in Fig. 1.

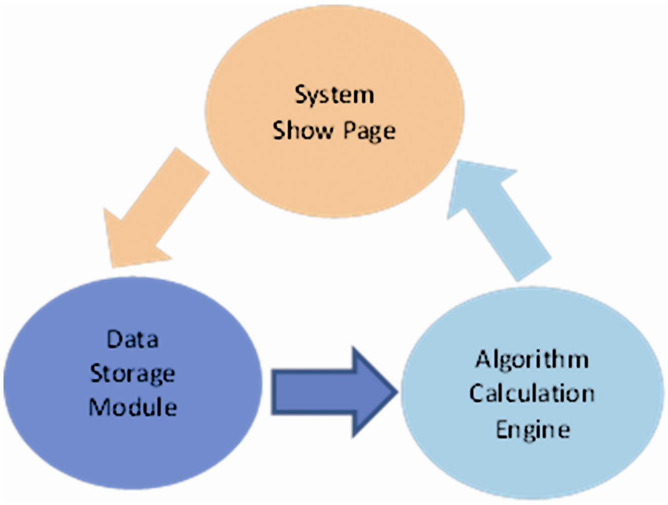


**Fig. 1.** Recommended system architecture

Algorithm calculation engine is the core part of the personalized recommendation system; better recommendation algorithm can calculate the user's potential interest information timely and accurately, provide good experience for users. The recommendation algorithm most widely used in actual production environment is Item-Base

Collaborative Filtering, aka IBCF. The main idea of IBCF is based on history rates of user for item, and generate item similarity matrix. Search out the item which is highest similarity with other items as the result of recommendations. From the perspective of computing, IBCF recommendation algorithm implementation process is divided into the following five steps:

1. **Item Rating Item Quantization**

When using IBCF algorithm to calculate the two items similarity, first, collecting the existing historical data from the database, quantify the rate of two items, Dimension is the number of users who has comment rating on an item, the value of dimensions is users' ratings for an item. Assuming that recommendation system contains $m$ users, collection is expressed as $U = \{U_1, U_2, \cdots, U_m\}$ and $n$ items, collection are expressed as $I = \{I_1, I_2, \cdots, I_n\}$. So the item $I_i$ corresponding rate vector of items can be represented as $R_i = \{R_{1i}, R_{2i}, \cdots, R_{ui}, \cdots, R_{mi}\}$. Among them, $R_{ui}$ expressions the rating of user $U_u$ to item $I_i$. The rating value can be viewed as the preference degree of user $U_u$ to item $I_i$.

2. **Similarity between Items**

The commonly used algorithm which to calculate similarity of two items include Cosine index similarity or Pearson Correlation Coefficient, etc. The following use Pearson Correlation Coefficient as an example, the Coefficient defined the Similarity through the linear relationship between two vector which corresponding to two rating for items, from the perspective of linear algebra is to compute the cosine value of between two vectors. Compared with other computing similarity method, it also take into account the difference of two mean rating for items, to eliminate the effects of differences between items, show user's authentic preferences for item. As shown in formula (1).

$$Sim_{ij} = \frac{\sum_{u \in N_i \cap N_j} \left(R_{ui} - \bar{R}_i\right)\left(R_{uj} - \bar{R}_j\right)}{\sqrt{\sum_{u \in N_i \cap N_j} \left(R_{ui} - \bar{R}_i\right)^2 \left(R_{uj} - \bar{R}_j\right)^2}} \tag{1}$$

$Sim_{ij}$ represents the similarity between the item $I_i$ and item $I_j$. $N_i \cap N_j$ expresses users intersection which both give the item $I_i$ and $I_j$ rate.

3. **Generate Project Similarity Matrix**

After calculating the similarity between the values of all project systems, to generate similarity matrix between projects, such as (2) shown in Equation.

$$\begin{bmatrix} Sim_{11} & Sim_{12} & \cdots & Sim_{1n} \\ Sim_{21} & Sim_{22} & \cdots & Sim_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ Sim_{n1} & Sim_{n2} & \cdots & Sim_{nn} \end{bmatrix} \tag{2}$$

Which $Sim_{ij}$ represents the similarity value between item $I_i$ and $I_j$ (where $I_i$ and $I_j$ belong to $I$), the value can be seen as the degree of similarity between items.

4. **Generate nearest neighbor set**

After obtaining the similarity between items, typically use two methods to select the nearest neighbor set for target item: one is a pre-set threshold similarity; the similarity between all the target items exceeds the threshold composition project nearest neighbor set V [11]. Another approach is to select the maximum K similarity value as a neighbor set items $V = \{I_1, I_2, \cdots, I_k\}$. Wherein $I_i \notin V$, and satisfies $Sim_{ik} > Sim_{ik+1} > \cdots Sim_{in}$, $n$ as the total number of items.

5. **Generate the final recommendation list**

According to the target items' nearest neighbor set of rates to predict the current user's score to the target item, the highest score before the election predicted a number of items as a result of recommendation to the current user. User rating prediction target item target item by nearest neighbor set score obtained, calculated as shown in Eq. (3).

$$R_{uj} = \overline{R_i} + \frac{\sum_{i \in V} Sim(R_{uj} - \overline{R_j})}{\sum_{i \in V} Sim_{ij}} \tag{3}$$

Which $R_{uj}$ represents the system predicts user $U_u$ to item $I_i$ rate. $R_{uj} - \overline{R_j}$ represents the average score of the project user ratings minus the project, it is to project the center received an average rating of (Mean-centering) treatment [12], to eliminate differences in scores between projects.

## 2.2 Hadoop Platform

Hadoop platform is one of the most popular solutions for big data problem. The Hadoop distributed file system and MapReduce programming model is the core of the distributed computing framework. It is provided by the Apache Software Foundation, aka ASF. Users can in the case of don't need to understand the underlying system details, use Hadoop organize computing resources to build their own distributed computing platform, Develop a distributed application, to make full use of cluster performance distributed and parallel processing massive data problem.

## 2.3 HBase Platform

HBase is a column oriented, scalable, and distributed No-SQL database, it can manage mass data which stored on thousands of server nodes efficiently and reliably. HBase is the open source implementation for Google's paper named Big-Table, and it has become the top project of the Apache Software Foundation. Compared with RDBMS, HBase's column oriented storage mechanism is suitable for storing sparse data. RDBMS is mainly suitable for transactional demanding situations, According to the theory of CAP, in order to achieve the strong consistency, need to synchronize by strict ACID transactions, this makes the performance of the system is reduced greatly in availability and scalability. HBase database is an eventual consistency mechanism, in the process of storage for transparent data segmentation, make store itself has a horizontal scalability and extensibility.

## 3 Personalized Recommendation System Algorithm Optimization

In the actual production environment, the number of users and projects in the personalized recommendation system often will be more than ten million levels, and the number of daily magnitude is also very large. Such large computing tasks, online real-time computing systems simply not load it in many cases, and use of off-line calculation also takes a long time to complete the update of similarity between the items, but user's demand for the system is more biased in favor of the real-time feedback. Therefore, to study and solve the problem of poor scalability recommendation system exists, it is very valuable.

Hadoop's distributed computing implementation is based on the MapReduce programming model, the basic idea is to split the entire data file into a number of blocks with a certain size, and store the data block in each storage node in the Hadoop cluster. When performing recommendation algorithm, split the entire calculation job into plurality Map subtask accordance with the number of data blocks, mapped the subtask to each computing nodes in the cluster for parallel computing. Map stage reads the data block file, generates intermediate key/value pairs, namely <Key, Value>, and writes the output to a local disk for persistent storage. After the completion of all Map stages, each computing node starts Reduce stage, read the output results of Map, after the reduction process outputs the final result. It is implementation shown in Fig. 2.

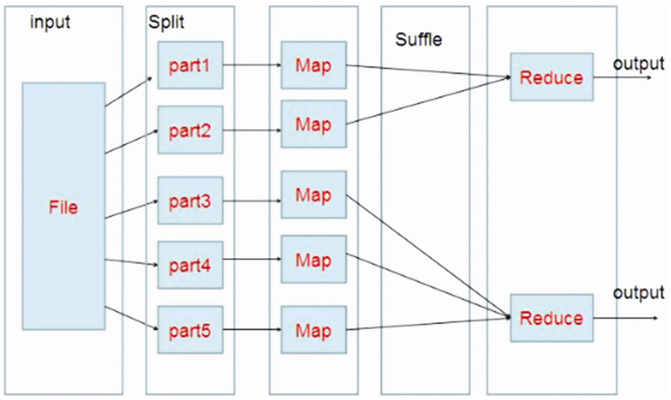


**Fig. 2.** MapReduce implementation process

According to the Sect. 2.1 implementation IBCF algorithm, now use the Hadoop MapReduce programming model to improve it. The process of calculating the predicted ratings of user for item will be divided into four MapReduce task is completed.

The first MapReduce task: Map stage reads the data block file with key-value pair <Key, Value>. After Map function processing, data UserID as Key, (ItemID, Rating) for the Value, i.e. <UserID, (ItemID, Rating)>. Reduce stage to perform a reduction process User ID as Key, the resulting generate User-Item scoring matrix. It is implementation shown in Fig. 3.

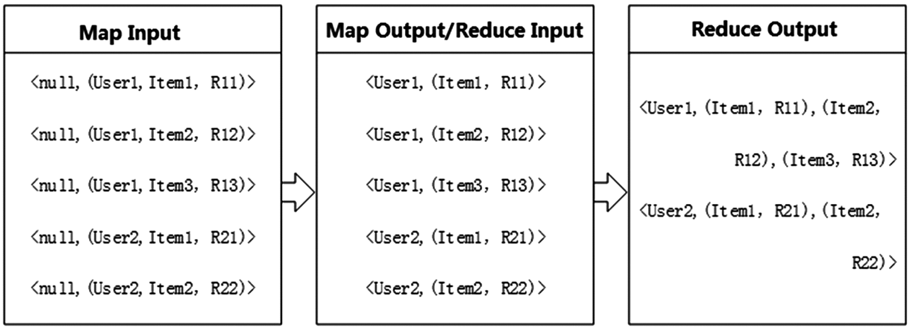

Map Input
<null, (User1, Item1, R11)>
<null, (User1, Item2, R12)>
<null, (User1, Item3, R13)>
<null, (User2, Item1, R21)>
<null, (User2, Item2, R22)>
Map Output/Reduce Input
<User1, (Item1, R11)>
<User1, (Item2, R12)>
<User1, (Item3, R13)>
<User2, (Item1, R21)>
<User2, (Item2, R22)>
Reduce Output
<User1, (Item1, R11), (Item2, R12), (Item3, R13)>
<User2, (Item1, R21), (Item2, R22)>

**Fig. 3.** Generate User-Item rating matrix

The second MapReduce task: Map stage reads the data block file, output to ItemID as the Key, (UserID, Rating) as the Value, i.e. <ItemID, (UserID, Rating)>. Reduce stage perform reduction processing ItemID as the Key to generate Item-User rating matrix. It is implementation shown in Fig. 4.

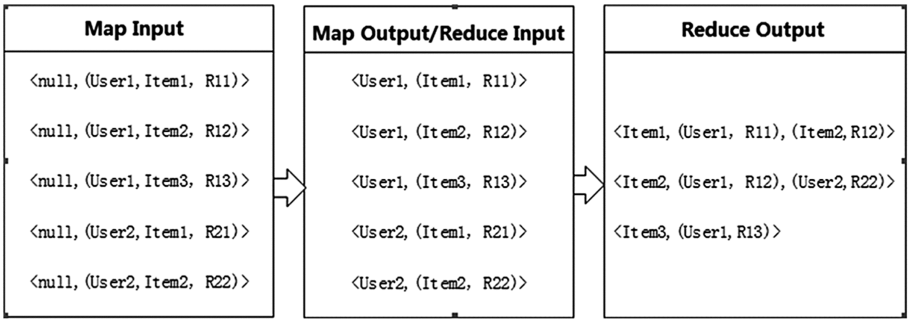

Map Input
<null, (User1, Item1, R11)>
<null, (User1, Item2, R12)>
<null, (User1, Item3, R13)>
<null, (User2, Item1, R21)>
<null, (User2, Item2, R22)>
Map Output/Reduce Input
<User1, (Item1, R11)>
<User1, (Item2, R12)>
<User1, (Item3, R13)>
<User2, (Item1, R21)>
<User2, (Item2, R22)>
Reduce Output
<Item1, (User1, R11), (Item2, R12)>
<Item2, (User1, R12), (User2, R22)>
<Item3, (User1, R13)>

**Fig. 4.** Generate Item-User rating matrix

The third MapReduce tasks: Map stage reads the Item-User Matrix produced by the last process, and output result use Item pair $(I_i, I_j)$ as the Key, the set of Rating as the Value. Reduce stage to calculate the similarity between items according to Eq. (1), the resulting similarity matrix as Item-Item (2) formula. It is implementation shown in Fig. 5.

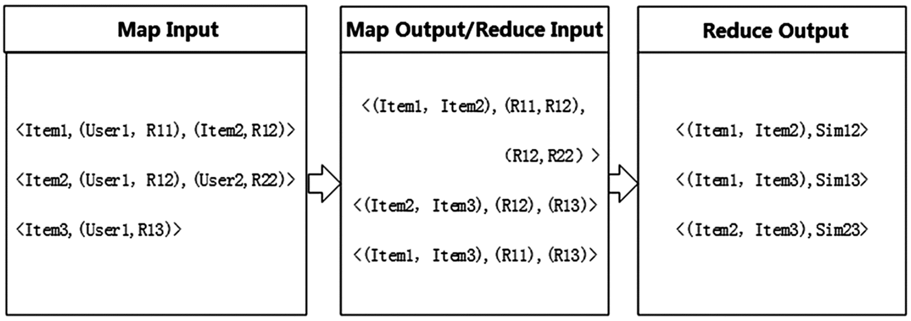

Map Input
<Item1, (User1, R11), (Item2, R12)>
<Item2, (User1, R12), (User2, R22)>
<Item3, (User1, R13)>
Map Output/Reduce Input
<(Item1, Item2), (R11, R12), (R12, R22) >
<(Item2, Item3), (R12), (R13)>
<(Item1, Item3), (R11), (R13)>
Reduce Output
<(Item1, Item2), Sim12>
<(Item1, Item3), Sim13>
<(Item2, Item3), Sim23>

**Fig. 5.** Generate similarity matrix

The fourth MapReduce tasks: Map stage reads the similarity matrix between items, select each item's top K highest similarity value as the nearest neighbor. According to Eq. (3) calculated prediction rate for each item. Reduce stage reduction process Map's result; select the top K highest rateing as the result of the final recommendation. It is implementation shown in Fig. 6.

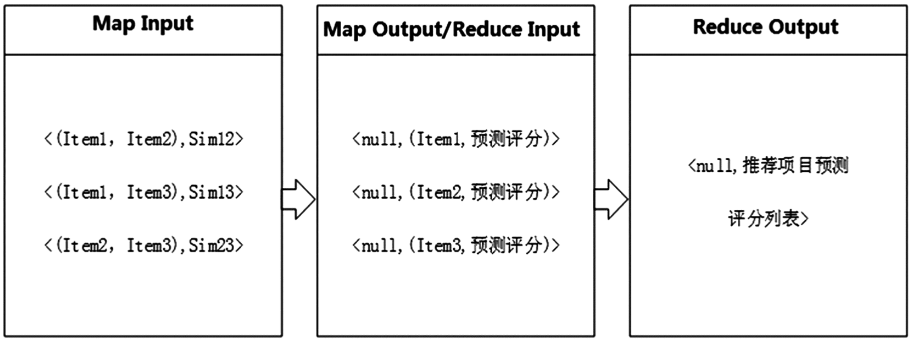


**Fig. 6.** Generate recommendation list

## 4 Data and Experiment

### 4.1 Experimental Data

The experimental data Movie-Lens is provided by the Group Lens Research project team is a free movie recommendation data, contents include: about ten million lines, data size is 252 MB, number of users is 71567, and number of film is 10681. The scope of user's ratings is [1, 5], the value of the rating on behalf of the evaluation of users.

### 4.2 Experiment and Result Analysis

This experiment to configure Hadoop cluster on 7 nodes, configure the Name-Node Server on the NO.01 and NO.02 machines, and configure Resource-Manager Server on the NO.03 machine. On the remaining 4 machines configure Data-Node Server and Node-Manager Server as storage nodes of HDFS and compute nodes of MapReduce. The configuration of each machine as follows: the CPU Model is Intel(R) Core(TM) i5-3470, the Memory is 4 GB, Hadoop version is 2.4.1, Linux version is CentOS-6.6.

#### 4.2.1 Data Storage Optimization Experiment

According to the characteristics of the data set, design the data storage model for RDBMS and HBase. The data model based on RDBMS storage, need to design three tables: the User table, Movie table and Rating table. Respectively, used to storing User's basic information, movies' basic information and the ratings of user for Movie. For example as the following User table, shown in Table 1.

**Table 1.** User table design

| UID | UName | UAge | UGender | … |
|---|---|---|---|---|
| ID1 | Name1 | Age1 | Gender1 | … |
| … | … | … | … | … |
| IDn | Namen | Agen | Gendern | … |

Among them, the field UID is the primary key of User table and can uniquely identify each row in the table. The field UName, UAge, UGender meaning the basic attribute of user, the data Namen, Agen, Gendern meaning the value of users' each attribute.

Use HBase to store the experimental data, just need to design one table only. This experiment based on HBase design the data storage model, as shown in Table 2.

**Table 2.** HBase data model design

| Row key | Time stamp | Column family:c1 | | Column family:c2 | |
|---|---|---|---|---|---|
| | | Info | Value | Rating | Value |
| $U_1$ | t6 | c1:Age | Value1 | | |
| | t5 | c1:Name | Value2 | | |
| | t4 | c1:Gender | Value3 | | |
| | t3 | | | c2:MovieID | $R_{ui}$ |
| $U_2$ | t2 | c1:name | Value1 | | |
| | t1 | | | c2:MovieID | $R_{2i}$ |

Among them, the U1 and U2 is the HBase database's RowKey, is equivalent to the primary Key of the RDBMS. The field c1 and c2 respectively HBase database's Column Family, and the Age, Name, MovieID in the equivalent of the columns for an RDBMS. The default value of Timestamp is the system time stamp, Can also be customized according to the business needs, each record has a corresponding timestamp.

During the experiment, selected 10M, 20M, 50M, 100M and 200M five groups of data sets of different sizes as the experimental data, using the above-described two types data model, the experimental data are being stored in the RDBMS database and HBase database, And measured different data models corresponding data storage file size. The results are shown in Fig. 7.

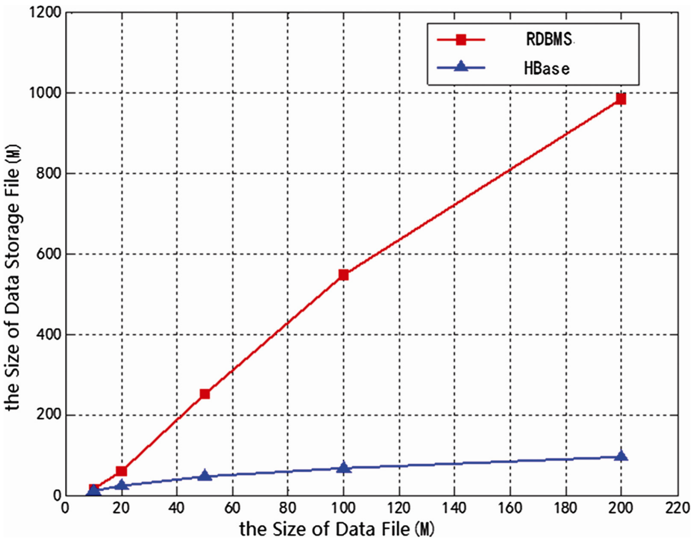


**Fig. 7.** Storage file size corresponding to different data models

Among them, Abscissa represents the different sizes of selected experimental data sets, in megabytes (MB) of units. Ordinate respectively represent the storage file size of RDBMS-based storage and HBase-based, in megabytes (MB) of units.

The above results show that, with the increasing size of the data set, RDBMS-based storage file size increased significantly, while HBase-based storage file size increase is relatively flat. This is because there are a lot of sparse data in the recommended system, namely a large number of meaningless null (Null value) exists. For these Null values, based RDBMS storage takes a lot of storage space, largely reducing the use of performance RDBMS database. The HBase column-oriented storage features for null values (Null value) does not take up any storage space, thus saving storage space and improved read performance, it is suitable for storing sparse data.

### 4.2.2 Recommended Algorithm Optimization Experiments

The first set of experiments to verify the performance of the Hadoop platform recommendation algorithm execution, gradually increase the number of nodes in the cluster, and record the response time of recommendation algorithm. First, use a single node test the time of calculate a user's predicted rate for all items, and then gradually increase the number of nodes from 2 to 6, and recorded all the time of calculate a user's predicted rate for all items. The results are shown in Fig. 8.

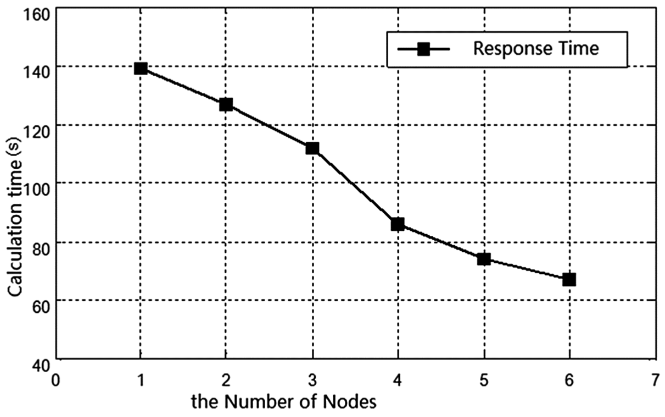


**Fig. 8.** System response time corresponding to different nodes

Wherein, Abscissa represents the number of nodes. Ordinate represents the corresponding system response time for different number of nodes, in seconds (s) as a unit.

The above results show that as the number of nodes increases, the computing time continue to decrease, indicating that the system performance continues to improve, but also found time decreasing amplitude is also declining because of the increasing number of nodes, the system do Map/Reduce operating system overhead required is also increasing.

The second set of experiments selected 10M, 20M, 50M, 100M and 200M five groups of data sets of different sizes as experimental data. Computing a user's predicted rate for all items, and record the recommendation algorithm's response time under different environmental. The results are shown in Fig. 9.

Among them, Abscissa represents the different sizes of selected experimental data sets, in megabytes (MB) as a unit. Ordinate represents the recommendation algorithm's response time for different modes, in seconds (s) as a unit.

The above results show that with the increase of the amount of data input, serial environment recommendation algorithm for memory resource consumption increases, resulting in decreased performance of the algorithm is not complete computing tasks. In a multi-node Hadoop cluster recommendation algorithm can complete large-scale data computing tasks, which fully shows that the use of based Hadoop parallel computing method can solve recommendation algorithm computing scalability issues.

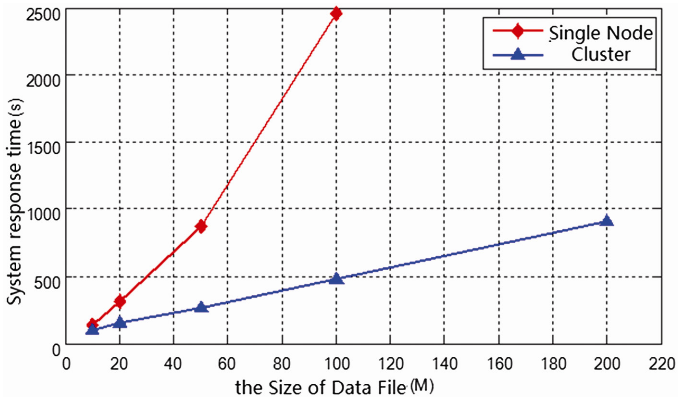
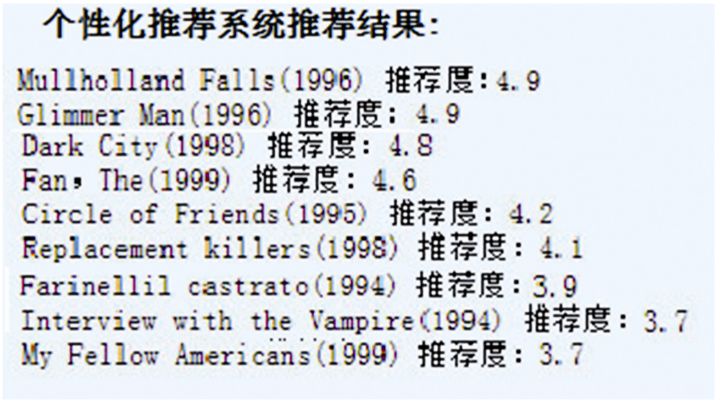


**Fig. 9.** System response time under different modes

### 4.2.3 Recommended System Results

Through the above detailed design of personalized recommendation system in data storage module and recommendation algorithm module. Article shows the implementation process of personalized recommendation system, basically reached the goal of personalized recommendation. Figure 10 shows the result of personalized recommendation system. Among them, the recommendation results in descending order according to the size of the recommended values, recommended values represent the recommended system predicted value for the target users' interests.



```
个性化推荐系统推荐结果:

Mullholland Falls(1996) 推荐度:4.9
Glimmer Man(1996) 推荐度: 4.9
Dark City(1998) 推荐度: 4.8
Fan, The(1999) 推荐度: 4.6
Circle of Friends(1995) 推荐度: 4.2
Replacement killers(1998) 推荐度: 4.1
Farinellil castrato(1994) 推荐度:3.9
Interview with the Vampire(1994) 推荐度: 3.7
My Fellow Americans(1999) 推荐度: 3.7
```

**Fig. 10.** Personalized recommendation system recommended results

## 5 Conclusion

Depending on the needs of users to quickly and accurately push the required information, provide personalized recommendation service is currently a hot research topic. In this paper, the current recommendation system in the data storage scalability and recommendation algorithm expansibility of both the problems, research and analysis of the IBCF algorithm, combined with Hadoop, HBase technology, design a suitable for large data environment data storage and processing solutions. Proposed based on Hadoop and HBase based personalized recommendation system, and use Group Lens Research project offers movies datasets, programming analysis and testing of the different size of the data file. Experimental results show that the proposed personalized recommendation system not only improves data storage scalability, and enhanced recommendation algorithm scalability.

## References

1. Cao, H., Fu, K.: Clustering collaborative filtering recommendation system search method. Comput. Eng. Appl. **50**(5), 16–20 (2014)
2. Fu, S.: Personalized information retrieval technology review. Inf. Theory Pract. **32**(5), 107–113 (2012)
3. Sarwar, B.: Sparsely. Scalability and distribute in recommender systems. University of Minnesota (2011)
4. Guopxia, W., Heping, L.: Personalized recommendation system overview. Comput. Eng. Appl. **48**(7), 66–76 (2012)
5. Ying, C., Wang, Z.: SVD feature: a tool kit for feature-based collaborative filtering. J. Mach. Learn. Res. **13**(1), 3619–3622 (2012)
6. Scarab, Katipos, G., Konstanz, J.: Incremental singular value decomposition algorithms for highly. In: Scalable Recommender Systems Fifth International Conference on Computer and Information Science, pp. 27–28(2012)
7. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2011)
8. Braes, J., Heckerman, D.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI 1998), pp. 43–52 (2012)
9. Miller, B.N., Albert, I., Lam, K., et al.: Movie-Lens unplugged: experiences with an occasionally connected recommender system. In: Proceedings of the Conference on Human Factors in Computing Systems, pp. 210–217 (2011)
10. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item to Item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2013)
11. Zeng, C., Xing, C.X., Zhou, L.Z., et al.: Similarity measure and instance selection for collaborative filtering international. J. Electron. Commer. **4**(8), 115–129 (2011)
12. Schafer, J., Konstan, J., Riedl, J.: Recommender systems in ecommerce. In: Proceedings of ACM E-Commerce, pp. 158–166. ACM Press, New York (2013)