Big Data Cleaning Algorithms in Cloud Computing

http://dx.doi.org/10.3991/ijoe.v9i3.2765

Zhang Feng [1, 2], Xue Hui-Feng[1], Xu Dong-Sheng [3], Zhang Yong-Heng[2],You Fei[2]
[1] Northwestern Polytechnical University, Xi'an, China
[2] Yulin University, Yulin, China
[3] Xi'an University, Xi'an, China

Abstract—**Big data cleaning is one of the important research issues in cloud computing theory. The existing data cleaning algorithms assume all the data can be loaded into the main memory at one-time, which are infeasible for big data. To this end, based on the knowledge base, a data cleaning algorithm is proposed in cloud computing by Map-Reduce. It extracts atomic knowledge of the selected nodes firstly, then analyzes their relations, deletes the same objects, builds an atomic knowledge sequence based on weights, lastly cleans data according to the sequence. The experimental results show that the cloud computing environment big data algorithm is effective and feasible, and has better expansibility.**

Index Terms—**big data, cleaning algorithms, cloud computing, data cleaning, Map-Reduce**

I. INTRODUCTION

With the development of Internet of Things, its technology has been widely used in to various fields, and has accumulated massive data [1]. Since the emergence of Cloud Computing, with the continuous development of science and technology and advance by academia and industry, the applications of Cloud Computing are going on developing. Cloud Computing is moving from theory to practice.

With the development of Cloud Computing, data center is also improved.Nowaday, data center is not only a site which manages and repairs servers, but also a center of many computers with high performance which could compute and store huge data. Currently there are proposed many cleaning algorithms of big data [2], that is mainly divided into regional object cleaning algorithm [3], the object cleaning algorithm based on information theory [4-5], based on discernibility matrix and on the basis of improved object cleaning algorithm [6] .Many scholars mainly study on how to deal with inconsistent decision table [7] and how to improve the efficiency of big data algorithm [4, 8-9].

The big data cleaning is the important way to resolve the massive data mining problem, and the big data cleaning algorithm combined the parallel genetic algorithm and co-evolutionary algorithm [10], to decompose of object cleaning task, which can improve the efficiency of big data algorithm. To this end, such big data cleaning algorithm that assume all the data can be loaded into the main memory at one-time, which are infeasible for big data. Cloud computing is a new business computing model that was proposed in recent years, it is the development of distributed computing, parallel computing and grid computing [11]. The pioneers of cloud computing is Google Inc. ,proposed a massive data storage and access capacity of large distributed file system GFS(Google File System) [12], and providing a handle massive data parallel programming mode of MapReduce, that provides a feasible solution for massive data mining [13]. Cloud computing technology has been applied in the field of machine learning [14], but there is still no real application to big data cleaning algorithm [15].

This paper has studied the Map-Reduce big data programming technology, analysis of existing big data cleaning algorithm, by analyzing the limitation of traditional structures of knowledge base， an extended tree-like knowledge base is built by decomposing and recomposing the domain knowledge, combined with MapReduce technical, the algorithms that suitable for large-scale data sets of big data computing equivalence classes is designed , and the big data cleaning algorithms are implemented in the cloud computing environment based on the Hadoop open source platform. The experimental results show that the algorithm not only has good scalability, but also able to handle the huge amounts of data.

II. CLOUD COMPUTING ALGORITHM FOR BIG DATA

Assuming that the decision table named T has n different values of decision object, the compatible object decision object values are mapped to 1..., n, set all incompatible object decision object value mapping for n +1. So we can set the decision table T can be composed by n+1 sub-decision table, that is D_1, D2,.., D_n, each sub-decision table contains the same class objects, the objects number are n^1,n^2,...,n^n, to this end, the decision table T is a compatible decision table. Assuming the object a have different object values, which is mapped to 1... , r. Set n_p^i is the number of objects where p is the mapping value of a condition object in the D_i. Obviously, the p object number is n_p^i ,so we can write

$$p\left(x_m \mid n_p^i\right)=\int p\left(x_m \mid x_{m-1},n_p^i\right)p\left(x_m \mid n_p^i\right)dx_{m-1}$$
$$=\int p\left(x_m \mid x_{m-1}\right)p\left(x_m \mid n_p^i\right)dx_{m-1} \qquad (1)$$

A. iscernibility objects calculation method in the cloud computing environment

A discernibility object is generated by two objects that they have different value of decision object and conditional portfolio object. If two objects decision value is different, and then the condition object a property value is also different, then a can identify these two objects, that is has the relative discernibility ability. The more number of objects that can recognize by an object, the stronger relative identification ability can be used for object discernibility number to measure relative discernibility ability.

In a compatibility decision tables T, $A \in C$, a can identify the object of the property of a pair and is given by

$$ObjP_q = \{< M, N > | f(M,q) \neq f(N,q)\} \quad (2)$$

In a compatibility decision table T, $Q \subseteq C$, the object set Q is able to recognize objects is given by

$$ObjP_Q = \{< M, N > | \exists q \in Q, f(M,Q) \neq f(N,Q)$$
$$, M \in D_i, N \in D_j\}$$
$$\text{and } 1 \leq i < j \leq N \quad (3)$$

Assuming U is imported from A that divided into r equivalence classes, set U/A= {A₁, A₂, …, Aᵣ} , the object combinations are mapped to 1, …, r. Where A is able to recognize the object of a number is calculated according to the following definition.

In a compatibility decision tables T, $A \subseteq C$, the object set A is able to recognize objects is given by:

$$ObjS_A^D = \sum_{1 \leq p < q \leq r} n_p^i n_q^j (v_s, v_1), (v_1, v_2), \ldots, (v_{i-1}, v_i) \quad (4)$$

It can be obtained by the definition (4), the discernibility objects calculation method is related to the exported equivalence class from A and D based on U, and then run cross-op operation, the calculation is relatively complex, so we can only calculate the discernibility objects pairs number in the memory based on the small-scale sets of the object set A. However, in the cloud computing environment, due to the large data of different equivalence class is stored in a plurality of nodes and files, the calculation of $ObjS_A^D$ involves many different equivalence classes, therefore, cannot be according defined to computing object number. How to quickly calculate the object identification becomes a key issue in data cleaning algorithm under the cloud computing environment of the number. Two calculation methods below focus on cloud computing environment can discernibility objects number.

As we all know, the set of objects A is able to distinguish between any two equivalence classes U/ A, it's showed that A has a certain amount of recognition ability. If A can only be divided into all the elements of an equivalence class, that A has the weakest recognition ability. Therefore, the relative identification ability of size $ObjS_A^D$ can use the object set A identification ability to calculate A.

In a compatibility decision table T, $A \subseteq C$, the object set A is able to recognize objects is given by:

$$ObjS_{U,A} = \sum_{1 \leq p < q \leq r} n_p \times n_q (v_s, v_1), (v_1, v_2), \ldots, (v_{i-1}, v_i)$$
$$(5)$$

In a compatibility decision table T, $A \subseteq C, c \in C \cup D$, where the new add identification ability of object c is defined sum size in A₁,A₂...,Aᵣ and are given by

$$ObjS_{U,A \cup \{c\}} \mathrm{I}\ ObjS_{U,A} =$$
$$ObjS_{A_1,\{c\}} \mathrm{U}\, ObjS_{A_2,\{c\}} \mathrm{U} \ldots \mathrm{U}\, ObjS_{A_r,\{c\}} \quad (6)$$

B. Big data cleaning algorithm in cloud computing environment

It can been find from the definitions (2) and (5) , object set A can identify the object to the object number $ObjS_A^D$ or not identification are needed to obtain through the calculation of equivalent to a number $ObjS$, but different equivalence classes can be parallel computation. Therefore, we can use the Map-Reduce big data programming techniques to handle large data.

In MapReduce programming framework, the user focuses on big data operation algorithm, write a Map function and Reduce function, realize the large-scale data big data processing. Specifically, the Map function is mainly complete different data blocks in the equivalence class, object and function of Reduce system with an equivalent number or calculation with an equivalence class is indiscernibility number. Based on different calculation methods of $ObjS_A^D$, given two kinds of data cleaning algorithm in the cloud computing environment.

In a compatibility decision tables T, $A \subseteq C$, where A is a necessary and sufficient condition of C relative to decision object D and is given by

$$\forall a \in A, \{ObjS_{A-\{a\}}^D \mathrm{I}\ ObjS_A^D\} < \{ObjS_A^D \mathrm{I}\ ObjS_C^D\} \quad (7)$$

Where $ObjS_A^D$ contains three algorithms that is Map function (algorithm 1), Reduce function (algorithm 2) and the main program (algorithm 3), which are described below.

```
Algorithm 1.  Map（Object keys, String values)
Input: the selected objects set A, candidate object $c$,
decision object D, an object value.
Output: <equivalence class, times>
Set
Equeal_Ac_Calss, Equeal_d_Class, Equeal_Acd_Class
is object set $\{ObjS_{A-\{a\}}^D\}$, $\{ObjS_A^D\}$ and $\{ObjS_C^D\}$ imported
equivalence classes
public Map（Object key, Text value){
 // Map Method
 For (Object a :{ $ObjP_q$ }{
 Equeal_Ac_Calss = Equeal_Ac_Calss +getObject (value, a);
      Equeal_d_Calss =getObject (value,b);
      context.write(Equeal_d_Calss, 3);

  }
 for (Object a: $ObjS_{U,A}$){
  Equeal_Acd_Calss = Equeal_Acd_Calss + getObject
   (value, a) +"";
   context.write(Equeal_Acd_Calss,1);

   }
 }
```

Algorithm 2. Reduce（String EquealsClass,Vector values）. // Reduce Method

Input: equivalence classes EquealsClass, vector

```
Output:〈EquealsClass，Number of occurrences〉
public Reduce(String EquealsClass ， Vector v){
    int totalnum＝0;
    for ( int k=1;k<v.size(),k++){
     totalnum=totalnum+v.get(k);
    }
  context.write(EquealsClass, new IntWritable
  (totalnum));
}
Algorithm 3. Main program $ObjS_A^D$.
Input: A consistent decision table T
Output: A cleaning table Cls
public void Objects(){
final JobConfconf = new JobConf(MapReduce.class);
final RunningJob job = JobClient.runJob(JobConfconf)
    $Cls = NULL$;
    While ( $ObjS_{U,A\cup\{c\}} \cap ObjS_{U,A}$ ! =NULL) {
  for (Object c: { $ObjS_{U,A\cup\{c\}} \cap ObjS_{U,A}$ }){
        Call for the Map function algorithm 1
        $Q_{value} = ObjS^D_{Cls\cup\{c\}} \cup ObjS_{U,A\cup\{c\}} \cap ObjS_{U,A}$
        $Cls = Cls \cup \{Q_{value}\}$;
      }
   }
  for (Object c: $Cls$ ){
        newReadTag. currentvalue + = valueStep;
  if ( newReadTag. currentvalue > = newReadTag. tag-
                          max) {
   newReadTag. currentvalue = newReadTag. tagmax;
                          };

       Call for the Reduce function algorithm 2
     $Cls = Cls \cap -\{c\}$;
      }
   }
  Output Cls;
}
```

Where algorithm 1 Map function calculates each data block equivalence class and the times, algorithm 2 set all data blocks in the same equivalence class were collected, and algorithm 3 were calculated identification ability, and the best candidate objects. According to the relative identification ability of each candidate objects, repeat the above process, until the calculated data cleaning.

C. *Big data parallel strategy*

Traditional parallel object cleaning algorithm is based on the assumption that all the data once loaded into memory, which is not suitable for large-scale data sets. The scale data is divided into a plurality of data segments in each task, parallel computing export candidate object set of equivalence classes, then to determine the optimum candidate based on calculated, each task cannot be identified or can be identified object number property, and ultimately to determine the best candidate .

TABLE I.
A CONSISTENT DECISION TABLE DATA

U	D	C_1	C_2	C_3
1	4	2	1	1
2	3	2	1	2
3	5	3	2	3
4	1	2	3	3
5	2	1	4	2

III. THE EXAMPLE ANALYSIS AND EXPERIMENTAL RESULTS

A. *The example analysis*

With a compatibility decision tables (Table 1) shows that the two algorithms proposed in this paper, are given in Table 1 all compatible object decision object values.

Assuming the table 1 is divided into two data segments, a data fragment contains 1 to 2 of objects, the first two data segments contains 3 to 5 objects. The following describes the algorithm to calculate the object on the number of processes.

$ObjS_A^D$ relative identification algorithm for computing the candidate object C_1 on process in Map stage, a data slice an object were generated objects derived equivalence classes C_1 <"C_1" 1> objects D export equivalence class <D 2, 1> and object collection {C_1, D} export equivalence class <"C_1 D 1 2 1> three <key, value> remaining object computing process is similar.

Reduce stage of objects C_1, a collection of objects D and objects {C_1, D} derived equivalence class merging, such as <"C 1 3" 1> and <"C_1 3" 1> merged into <"C_1 3 ", 2>;

<"C_1", 1>, <"C_1", 1>, <"C_1 2"> Merge <"C_1 2" 3>; equivalence class of the remaining properties export merger similar. Ultimately, the number of objects C_1 relative identification 6.

B. *Experimental result*

This section is analyzed the evaluate performance running time of speedup and scaleup in the cloud computing environment, used the $ObjS_A^D$ parallel operation strategy algorithm. The test performance use artificial data sets Obj1, Obj2, Ojb3 and Obj4. Table 2 lists the characteristics of the different sets of data. Using the open source cloud computing platform hadoop0.20.2 and java1.7 in 10 ordinary computers build cloud computing environment to experiment, where one master station and nine slave nodes.

Firstly, from the data cleaning algorithm running time comparison, in two small data sets Obj1 and Obj2 of $ObjS_A^D$ was compared three kinds of data cleaning algorithm. Can be seen from Figure 1, the small data set would not be appropriate to use the MapReduce technology, and the use of data and task parallelism, data cleaning method than using only data-parallel algorithm running time is shorter.

Secondly, Obj1 ~ 5 five data sets, respectively, in a four node test run time is shown in Figure 2.

TABLE II.
THE CHARACTERISTICS OF THE DIFFERENT SETS OF DATA

Data set	Objects	Condition values	Decision values
Obj1	35000000	12	3
Obj2	5000000	32	3
Obj3	45000000	23	4
Obj4	15000000	22	5
Obj5	65000000	2000	12

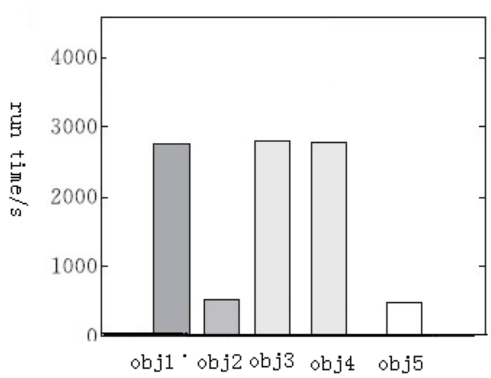

Figure 1. Cleaning algorithm running time comparison

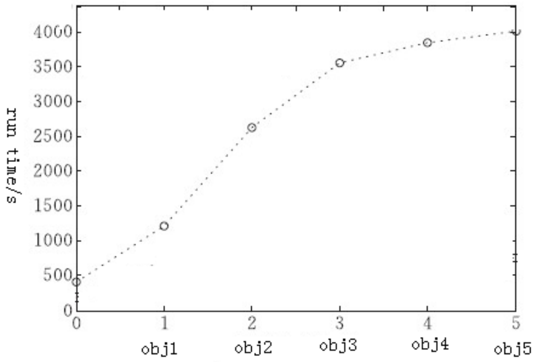

Figure 2. Running time comparing times object selection

The existing data cleaning algorithm used to calculate the minimum object cleaning, but only on a small data set cleaning, big data were randomly divided into several sub-decision tables, then calculate the number of positive region, respectively, for each sub-decision table select the optimal single candidate objects, and repeat the process in order to gain cleaning. However, for inconsistent decision table, the method does not guarantee to each decision table of computing positive region and calculation of the decision table is region are equivalent, because each decision table of computing positive region is not the exchange of information. And, it is also unable to larger subset of decision table cleaning. The decomposition of large-scale data by using Map-Reduce technology, cleaning of each data partition, then merge the cleaning of each data slice, add other necessary candidate objects, finally remove redundant objects. However, this method combined candidate cleaning may be larger set of condition objects become very difficult to remove redundant objects.

In order to solve the problem of data cleaning of large data sets, this paper are divided on the large-scale data by using Map-Reduce technology, equivalence calculation of different candidate object set for each data partition, and then summarize the equivalence classes, calculate the candidate object relative indiscernibility or identification of the object, select the optimal single candidate object, iteration of this process, until the cleaning. The proposed algorithm based on data parallel strategy, so that it can greatly save the MapReduce job start and scheduling time, thereby improving the data cleaning algorithm efficiency under the cloud computing environment.

IV. CONCLUSIONS

The existing data cleansing algorithms through improved sorting algorithm or using better data to quickly calculate the equivalence class, the time complexity is reduced, but can only be dealt with in the small data sets of main memory. This dissertation focuses on Cloud Computing data center, policy of replica, and scheduling mechanism, and makes deep research around such issues. It not only proposes network structure of data center, policy of replica and scheduling algorithms, but also provides a strong foundation for other related researches on Cloud Computing in the future. For large-scale data sets data cleaning, analyzed of large data cleaning algorithm can be parallelized operation, based on the knowledge base, a data cleaning algorithm is proposed in cloud computing by Map-Reduce. It extracts atomic knowledge of the selected nodes firstly, analyzes their relations, deletes the same objects, builds an atomic knowledge sequence based on weights, discuss and achieve a variety of data-parallel strategy, and experiments on ordinary computer cluster using Hadoop. Experimental results show that the data cleaning algorithm has better acceleration than able to handle large data sets.

REFERENCES

[1] Yu HJ, Liu YQ, Zhang M, Ru LY, Ma SP, "Research in search engine user behavior based on log analysis", Journal of Chinese Information Processing, 2007, 21(1):109−114 (in Chinese with English abstract).

[2] Wang, Huaibin, "*Virtual machine-based intrusion detection system framework in cloud computing environment*", Journal of Computers, v 7, n 10 SPL.ISS. P 2397-2403, 2012.

[3] Wang, Xiaoli, "Energy-efficient task scheduling model based on MapReduce for cloud computing using genetic rithm" ,Journal of Computers (Finland), v 7, n 12, p 2962-2970, 2012.

[4] Wu, Lijuan, "Realization of quadrilateral mesh partition and optimization algorithm based on cloud data" ,Journal of Computers, v 6, n 12, p 2519-2525, 2011. http://dx.doi.org/10.4304/jcp.6.12.2519-2525

[5] Yang, Tong, "Mass data analysis and forecasting based on cloud computing", *Journal of Software, v 7, n 10, p 2189-2195, 2012.*

[6] Zhang, Gang, "Data dependant learners ensemble pruning" *Journal of Software, v 7, n 4, p 919-926, 2012.*

[7] Junxiu, An, "*The demonstration of cloud retrieval system mode*", Journal of Software, v 6, n 2, p 249-256, 2011.

[8] Ji, Changqing, "*Big data processing in cloud computing environments*", Proceedings of the 2012 International Symposium on Pervasive Systems, Algorithms, and Networks, I-SPAN 2012, p 17-23, 2012.

[9] Cheng, Hongbing, "*Identity based encryption and biometric authentication scheme for secure data access in cloud computing*", Chinese Journal of Electronics, v 21, n 2, p 254-259, April 2012.

[10] Agrawal, Divyakant, "*Big data and cloud computing: New wine or just new bottles*", Proceedings of the VLDB Endowment, v 3, n 2, p 1647-1648, September 2010.

[11] Kohlwey, Edmund, "*Leveraging the cloud for big data biometrics: Meeting the performance requirements of the next generation bi-*

ometric systems", Proceedings - 2011 IEEE World Congress on Services, SERVICES 2011, p 597-601, 2011.

[12] Gao Hong-Wei, Yu Yan-Jun, Chai Feng, Cheng Shu-Kang."Position sensorless control of interior permanent magnet synchronous motor based on carrier frequency component method", Proceedings of the Chinese Society of Electrical Engineering, v30, n18, pp.91-96, 2010.

[13] Si Ji-Kai, Wang Xu-Dong, Yuan Shi-Ying, Ma Xing-He,Chen Hao, "Non-linear model establishment and steady state characteristics analysis of permanent magnet linear synchronous motor", Journal of the China Coal Society, v 35, n 2, pp. 343-348, February 2010.

[14] Y.W. Zhu, D.S. Kim, D.H. Kooa, Y.H. Cho, "Optimal design of a double-sided slotted iron core type PMLSM with manufacturing considerations", Applied Electromagnetic Engineering: Magnetic Superconducting and Nano Materials, vol.670, pp. 235-242, June 2011.

[15] Wei Hua, Gan Zhang, Ming Cheng, Jianning Dong, "Electromagnetic performance analysis of hybrid-excited flux-switching machines by a nonlinear magnetic network model", IEEE Transactions on Magnetics, vol. 47, pp.3216-3219, October 2011. http://dx.doi.org/10.1109/TMAG.2011.2154377

AUTHORS

Zhang Feng received the MS degree in Computer science from Xidian University in 2009. Now he is a PhD of Northwestern Polytechnical University. He is currently a associate professor at Yulin University. His research interests are in the areas of Cloud integrated manufacturing technology, the modeling of complex systems, the Internet of Things applications. (e-mail: tfnew21@sina.com).

Hui-Feng Xue received the PhD degree in Water resource economics from Xi'an Polytechnic in 1995. He is currently a professor in Northwestern Polytechnical University. His research interests are in the areas of the modeling of complex systems, Simulation and performance evaluation, management, systems engineering, energy and environmental systems engineering, computer control, intelligent control, network control. (e-mail: xhf0616@163.com).

This work is partially supported by the Defense basic research projects in 2013 #A0420131501, Funding Project for Department of Education of Shaanxi Province in 2013 #2013JK1167, Research and Cooperation Project for Department of Yulin city in 2012 #2011SKJ09. Received 17 January 2013. Published as resubmitted by the authors on 16 May 2013.