http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (23): 7345-7352, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.7345.7352

Research of Parallel Decision Tree Algorithm Based on Mapreduce

[1,2]Yuwan Gu, [1,2]Guodong Shi, [2]Huanhuan Cai, [2]Yan Chen and [2]Yuqiang Sun
[1]College of Electronic and Information Engineering, Jiang Su University, Jiang Su, Zhenjiang, 212013, China
[2]International Institute of Ubiquitous Computing, Chang Zhou University, Jiangsu, Changzhou, 213164, China

Abstract: Traditional decision tree algorithm has been unable to solve large-scale data mining; this study presents a parallel decision tree classification algorithm based on MapReduce. The algorithm uses the attribute set dependence as a test attribute selection criteria to avoid shortcomings of the ID3 algorithm ,which is difficult to remove noise, the relationship between attributes is not close enough, so using the MapReduce model to solve large-scale data mining problem. Verified by an example: decision tree algorithm based on MapReduce can handle massive data classification problem and has better scalability and higher efficiency of classification.

Key words: Data mining, massive data, ID3 decision tree, MapReduce

INTRODUCTION

The decision tree technology is a key technology of data mining algorithms; decision tree classification algorithm has been the research hotspot. In recent years, the study of the decision tree mining algorithm is mainly focused on improving the mining algorithms. These algorithms improve the effectiveness of the system of mining to a certain extent, but capacity of mining system for data processing is far enough. With the rapid development of information technology, the amount of data has grown exponentially, it is more important for massive data mining. Parallel mechanism is an effective way to solve massive data mining. Current mainstream parallel programming model are (in Message Passing Interface, MPI) (Hu and Wang, 2008), parallel virtual machine based on message passing, PVM (Paral lel Virtual Machine,) (Xiong *et al*., 2008), OpenMP based on shared storage(Wang *et al*., 2009). the biggest advantage of PVM' is the flexibility and is good at interaction fault tolerance and transplantation of heterogeneous platforms, while MPI is simple and easy-to-use, OpenMP provides a powerful compiler directives and library functions, but the three parallel abstraction is not high. Later, a skeleton-based parallel programming model is emerged (Zhu and Wan, 2009), it has a high level of abstraction and implicit internal implementation details, but for complex application problems to achieve combination of the skeleton is a troublesome thing,. Therefore, this paper the model of MapReduce proposed by Google, which can handle large-scale data (greater than 1TB) (Li *et al*., 2011), the model is highly abstracted into two functions; Map function and Reduce function for merging decomposition task. It can reduce the complexity of the design of parallel program, it encapsulates the operations, such as Data partitioning, tasks allocation, parallel processing primarily from the bottom,, the user are only needed to design parallel computing tasks.

Now there are some parallel mining algorithms, such as literature (Zhu *et al*., 2011; Yang, 2010) proposed a clustering method program based on the Hadoop platform, literature (Cheng and Chen, 2011) studied parallel genetic algorithm. these algorithms all have strengths and weaknesses and has been applied for research, but relatively few studies of the core of the decision tree algorithm-ID3 algorithm, so this article studies parallel decision tree algorithm. Based on MapReduce on the basis of ID3 algorithm theory.

PARALLEL STUDIES OF THE DECISION TREE ALGORITHM

There are many parallel research on decision tree classification algorithm, in conclude, there are mainly the following two ways: according to different method of data partitioning, they can be divided into: Dynamic data partitioning and static data partitioning; according to different program design, they can be divided into the main mode and peer-to-peer mode.

Method of data partitioning

Method of dynamic data partitioning: The traditional method of Dynamic data partitioning is fragmentation based on the task, first training set is stored on the host

Corresponding Author: Yuqiang Sun, ChangZhou University International Institute of Ubiquitous Computing, Jiangsu, Changzhou213164, China

processor, hypothesis there are m processors, training assembly be divided equally among the m sets processor. Then constructing a decision tree on m processors sin parallel. This approach is all the processor will participate in the distribution, it will make the load increase. Improved method is the main treatment will save an idle processor queue task, asany processor can apply for an idle processor to the main treatment agent handling to send signals to the live processor finished the task, then the host processor put the processor into the idle queue. Although improved methods can solve the load problem, but it will increase the time of the data movement and communication between the processors.

Method of static data partitioning: In order to alleviate shortcomings of the frequent moving and interprocessor communication of dynamic data partitioning, a static data partitioning method appears. Method of static data partitioning is divided into horizontal data partitioning and longitudinal data partitioning.

Horizontal division: Horizontal division is a division of the level of the training set, if the size of the training set is M, the number of processors is m. the scale which each processor processes data set is N / m and the calculation of the split point and the division of data sets can well be parallel. The time of traversing the data will be reduced, but a larger amount of traffic between processors will be required.

Lengthwise division: The vertical division is one or several of the complete lists of attributes assigned to a separate processor for processing, each processor will complete the process of one or more attributes of the split information. This process is executed in parallel. If the training set has N attributes, the time complexity of the longitudinal division 1 / N times smaller than the serial's and also significantly reduce the interprocessor communication.

Programming mode

Master-slave mode: Master-slave mode selects a processor as the host processor, the rest are from the processor, the host processor assign tasks to each processor, each processor computing division in parallel and sends the result to the master, the master combined result of each processor, the main processor identify the best split point form a hash table and then distributed to each processor, each processor divides subset on this machine according to the hash table. In this way load will be balanced communication exists only seeking the best attributes between nodes, so communication cost is also small.

Peer-to-peer mode: Peer-to-peer mode, all processors are peer-to-peer, not the master-slave communication between the processors can complete the construction of the decision tree. This way will consume a large amount of communication time; the hash table requires each processor must store information, so it will waste memory.

MAPREDUCE TECHNOLOGY

MapReduce is a software framework proposed by Google, is the basic computational model of a variety of cloud computing platform, the main function is the processing of massive data. Massive data set is divided into small data set to several computers for processing, in order to achieve parallelism, MapReduce sees data as a series of (key, value) pairs and simplifies data in two stages by Map mapping and Reduce Statute. The MapReduce computing processes shown in Fig. l, input, it is made of four parts the input, the Map task,the Reduce task, the output:

- Input. The MapReduce library will divide input file into the same size of M Split (film), the size of the piece is determined by the user and Split information is stored in a distributed file system (Hadoop Distributed File System, HDFS), taskswill be submitted to each processor, identify idle processors and assign sub-tasks for them (M Map subtask, R Reduce subtasks), place them in the queue
- Map tasks. The Map sub-tasks assigned can gain task-related data from the HDFS then <key, Value) pairs will be generated, the intermediate results of the Map function is written to the local disk; the intermediate results of Map function will be divvied into R districts through the partition function the location information of the local disk will be sent to Reduce subtasks node

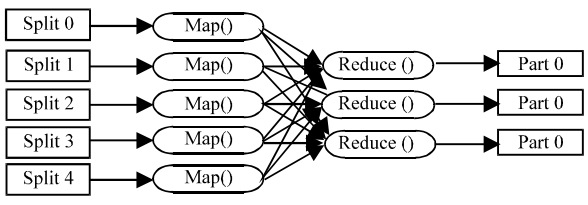

Fig. 1: MapReduce computation processes

- Reduce tasks. Reduce subtasks gain intermediate data from the location information and will merger the same key<key, Value) and sort them by value, combined the result is passed to the Reduce function Reduce function output the result to the output file
- Output. Completion of the Map and Reduce subtasks, JobTracker will return R copies of Reduce to the client program, the client program treat according to the results

STUDY OF PARALLEL ALGORITHMS OF ATTRIBUTE SET DEPENDENCE BASED ON THE MAPREDUCE

Decision tree algorithm based on attribute set dependence: Through research and analysis of the ID3 algorithm, to solve its shortcomings which are difficult to remove for noise and interrelation between attribute is not enough, we propose attribute set dependence. Attribute set dependence is based attributes reduction on considering properties interdependence, thus removing redundant attributes to simplify the structure of the decision tree.

The dependence of Decision attribute Q to a single attribute is called attribute dependence and the dependence of decision attribute Q to attribute of conditional attributes is called the set of attributes dependence. The Equation to solve the attribute set dependence remains Eq. 1:

$$k = r_p(Q) = \frac{card(POS_p(Q))}{card(Q)}$$

where, supposing K = (U, R) is an approximate space, P and Q as an equivalence relation family gathers on the global U and P, Q∈ R, if all types of Q can be defined by the types of the P, Namely the P can get the Q , then Q depends on P, denoted by P => Q. Or that knowledge Q depends on knowledge P with dependence k (0 = k = 1) only when:

Algorithm pseudo code.
The Function ASDM-Dtree (C: candidate attribute set, T: a training set) returns a decision tree.
Enter the training sample set T, the condition attribute set C.
Output a decision tree.
{
(1) Create a root node Root;
(2) If T is made of record with the same category attribute value, returns a single node with this value;
(3) If R is empty, returns the root as a leaf node and mark root as the largest class of appearing in record T;
(4) For each condition attribute set C in T
ASDM (C); / / function ASDM (C): attribute set dependence of Ci is calculated by the formula (1);
All object sorts according to the set of attributes demanded for solving / division of equivalence classes
nCount = 0 / / { equivalence class of the set of condition attribute } n{ equivalence class of the set of decision }, the initial 0
if the values of the attribute set of all objects are the same
Count = nCount + number of such objects
end for
return nCount / / total number of objects
k = nCount / the total number of decision attribute set / / return the dependence of each equivalence class
end ASDM
(5) Test(which is the current test attribute of Root) = Ci with the biggest dependence of attribute set in candidate attribute ;
(6) delete Ci from R, form a new candidate attribute set C ';
(7) returns a tree, its root is tagged for T ,and its branches are marked as d1, d2, ... dm;
8) for each values of Test = C '
{
grow a new child node from the node Root;
IF new leaf node corresponds to the subset of samples T 'is empty
Delete this node:
ELSE
ASDM-Dtree (C ', T' m);
}
}

Mapreduce-based decision tree algorithm thought: Time complexity of decision tree algorithm based on Attribute set dependence is O (n), in the case of larger data, the traditional serial algorithm will take a long time, not to mention for large data sets and parallel program can be a good solution to this problem. During the decision tree construction, the choice of the test attribute wastes a lot of time and is the most critical, attribute calculation is parallel , decision tree algorithm based on Attribute set dependence can be used in the MapReduce platform. This algorithm, using the master from the programming mode and static data partitioning method for large-scale data processing, to quickly and effectively pick out the best attributes. The process includes two parts: (1) to achieve the pre-processing of the data with Job 1, to construct decision tree algorithm. With Job 2.

Mapreduce-based decision tree algorithm processes: The Introduction of decision tree algorithm based on attribute set dependence, in order to calculate the dependability of attribute sets more easily, for each condition attribute and decision attribute, first obtain their corresponding property values and store them in the array A [m] [n], referred to as the number of values ??recorded matrix, m and n for the matrix rows and columns, respectively, for Job 1 MapReduce task does the data preprocessing and generates (attribute set, the attribute sets rely degrees) this key-value pairs, which are as input of function of Job 2Map, Fig. 2 is a detailed flow of the Job 2 tree construction.

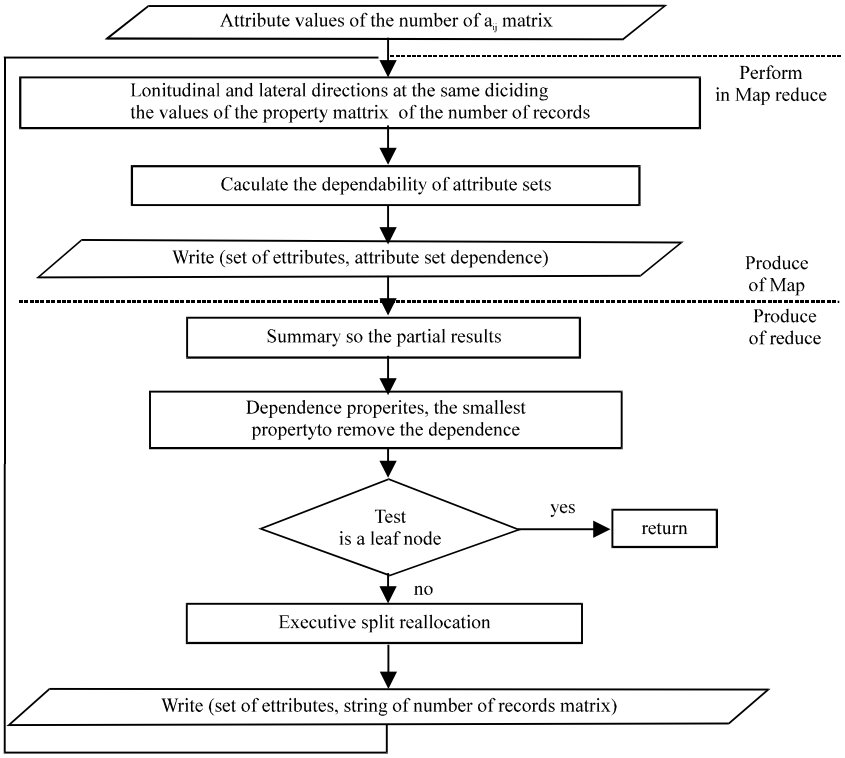

Fig. 2: Flowchart decision tree algorithm based on MapReduce

Specific achieve of attributes set dependence based on the mapreduce algorithm: The algorithm designs Job class to implement the main process of Fig. 2, the Job class contains Job-Mapper class and Job Reduce class. At the same time, the algorithm combines itself with algorithms based on attribute set dependence.

Job class, this class is as an initialization entry of program through MapperClass Class and ReduceClass of class methods to set Job_Mapper class and Job_Reduce class, which respectively acted as a function of the realization of the Map, Reduce class. Job_Mapper.

This class is used to implement the Map process. Firstly, read output of Job1 from HDFS, <(set of attribute names), (the value of the number of records matrix)> as data object will be read. Attribute set dependence of the data object will be calculated and the result will form output mode of the key values of <key, value), which key is (attribute name), value is for attribute set dependence of the current data object.

The pseudo code of Map process is as follows:
Input <key, value).
Output <key, value>, value is attribute set dependence.
(1) read the value of value
(2) Value '= ASDM (value)
/ / ASDM ()Find the attribute set dependence using Eq. 1
(3) Context. Write (key, value ')
(4) End

Reduce class. This Class implements the process of Reduce, firstly getting partial results from each node of Map process, then summarized this outcome, seeking the maximum value of the test attributes and finally realize the property division, then repartiting data sets and writting the result back.

Input to the output of <key, value), the same with Map.
The output <key 'value'):
(1) Create a node N
(2) For (value) seeking the maximum value of attribute set dependence, saving it in max and let node N is equal to the condition attributes with the biggest attribute set dependence
Call MakeTree (N, T) to build a node, do division for each branch of the non-leaf node.
Context. Write (key ', value'), 'value of key is each branch node of the current node N.
End

CASE STUDY

A foreign recruitment, for example, foreign companies oriented-society recruit employees both the recruitment of graduating students also recruit experienced person, we summarize a few of the more important factor in the many requirements: political landscape, work experience, practical ability, professional achievements foreign language proficiency, academic qualifications, to both N

Table 1: Foreign recruitment conditions

U	Political landscape	Work experience	Practical ability	Professional achievements	Foreigen language proficiency	Educational background	Probability of enter to foreign companies
1	Party members	Little	General	Good	Don't pass	Undergraduate	Little
2	Masses	Little	Strong	General	Don't pass	Bachelor degree or above	Big
3	Party members	Little	General	Good	Pass	Undergraduate	Little
4	Masses	Little	Strong	General	Pass	Bachelor degree or above	Big
5	Party members	Rich	General	Good	Don't pass	Undergraduate	Little
6	Masses	Rich	Strong	General	Don't pass	Bachelor degree or above	Big
7	Party members	Rich	General	Good	Pass	Undergraduate	Big
8	Masses	Rich	Strong	General	Pass	Bachelor degree or above	Big
9	Members	Little	Strong	Good	Don't pass	Undergraduate	Little
10	Members	Little	General	General	Don't pass	Undergraduate	Little
11	Members	Rich	General	General	Pass	Undergraduate	Big
12	Party members	Rich	General	General	Pass	Bachelor degree or above	Big
13	Members	Rich	Strong	Good	Don't pass	Undergraduate	Big
14	Party members	Rich	Strong	Good	Don't pass	Bachelor degree or above	Big
15	Masses	Rich	Strong	General	Pass	Undergraduate	Big
16	Members	Little	Strong	General	Pass	Bachelor degree or above	Big

and P represent the small and large probability of entering to foreign companies. The final statistics as shown in Table 1.

In order to calculate the dependability of attribute sets more easily, for each condition attribute and decision attribute, first obtain their corresponding property values and store them in the array A [m] [n], referred to as the number of values recorded matrix. political landscape value of (A) [1,2,3] represent the masses, delegates and party members; work experience (B) value of [1,2] represent small and rich; practical ability (C) value of [1,2], respectively, on behalf of the general and strong; xprofessional achievements (D) value of [1,2] represent general and excellent;; foreign language proficiency (E) value of [1,2] represent, but customs and customs; qualifications (F) value of [1,2] trepresent undergraduate degree and above; probability into foreign companies value of (G) [1,2] represent small and large.

First use map reduce for the entire training set of horizontal and vertical division, the size of the division decided by the users, here training set constituted by the condition attribute will be divided into two parts in horizontal, one part is 1-8 and the other part is 9-16, average longitudinally divided into two parts, part one is A, B and C; another part is he D, E, F,. So the entire training set is divided into four parts. Then calculate the set of attributes of the various parts of the decision attribute dependence parallelly, each part is then assigned a Map task Map function will analyze the intermediate key-value pairs (key, value) and cache them to memory and the key is property set ,the value is the attribute set dependence. Then keys cached in memory will be written to a local disk through partition function\periodically and finally sent to the Reduce tasks, Reduce subtasks gain intermediate data from the location

Table 2: Matrix of values of the number of records

U	A	B	C	D	E	F	G
1	3	1	1	2	1	1	1
2	1	1	2	1	1	2	2
3	3	1	1	2	2	1	1
4	1	1	2	1	2	2	2
5	3	2	1	2	1	1	1
6	1	2	2	1	1	2	2
7	3	2	1	2	2	1	2
8	1	2	2	1	2	2	2
9	2	11	2	2	1	1	1
10	2	1	1	1	1	1	1
11	2	2	1	1	2	1	2
12	3	2	1	1	2	2	2
13	2	2	2	2	1	1	2
14	3	2	2	2	1	2	2
15	1	2	2	1	2	1	2
16	2	1	2	1	2	2	2

information, will have the same key of the key, the Value) sort of the merger and then Reduce the result is returned to the client program and compare the size of each attribute. the decision attribute we want will have the biggest Attribute set dependence t but also can solve for minimum attributes, attribute set dependence of 0 and contains the minimum attribute, the minimum attribute is redundant attributes, you can directly removed. Repeat the above steps until can not compare the size of the properties, until the decision attribute can continue until there is no decomposition. The final decision tree constructed will be both simple and accurate:

Step 1: The training set is divided into four sub-set of the matrix shown in Fig. 2, it is the value of the number of records, red and black thick lines by the middle of four parts

Step 2: Parallel calculate dependence for each subset of the set of attributes

The first part of the attribute dependency is calculated as follows:

- The equivalence partitioning of Conditional attribute set {A, B, C}:

$$U/\{A, B\} = \{1,3\}\ \{2.4\}\ \{5,7\}\ \{6,8\}$$

$$U/\{A, C\} = \{1,3,5,7\}\ \{2,4,6,8\}$$

$$U/\{B, C\} = \{1,3\}\ \{2.4\}\ \{5,7\}\ \{6,8\}$$

- Equivalence class division of decision attributes:

$$U/\{G\} = \{1,3,5\}\ \{2,4,6,7,8\}$$

- {Equivalence class of condition attribute set} n {equivalence class of decision attribute} the number of elements of the set of attributes:

The dependence of attributes set of G on {A, B}, for example:

$$U/\{A, B\}\ n\ U/\{G\} = \{1,3\}\ \{2,4\}\ \{6,8\}$$

$$Card\ (POSC\ (G)) = 6$$

- The dependence of attributes set of G on {A, B} is 6/8

Similarly too:

- The dependence of attributes set of G on {A, C} is 0
- The dependence of attributes set of G on {B, C} is 6/8

The other three parts of the same, respectively, the dependence of the set of attribute obtained as follows:

The second part of the attribute dependency is calculated as follows:

- The dependence of attributes set of G on {D, E} is 6/8
- The dependence of attributes set of G on {D, F}is 4/8
- The dependence of attributes set of G on {E, F} is 6/8

The third part of the attribute dependency is calculated as follows:

- The dependence of attributes set of G on {A, B} is 5/8
- The dependence of attributes set of G on {A, C} is 4/8
- The dependence of attributes set of G on {B, C} is 1

The fourth part of the attribute dependency is calculated as follows:

- The dependence of attributes set of G on {D,E} is 0
- The dependence of attributes set of G on {D, F} is 3/8
- The dependence of attributes set of G on {E, F} is 5/8

Third step is performed by two functions, Map and Reduce:

Step 1: Input of the map data:

The input data of Map1

Key1	Value1
{A,B},{A,C},{B,C}	0

The input data of Map 2

Key1	Value1
{D,E},{D,F},{E,F}	0

The input data of Map 3

Key1	Value1
{A,B},{A,C},{B,C}	0

The input data of Map 4

Key1	Value1
{D,E},{D,F},{E,F}	0

Step 2: Output of map and input of combine

Output of map1

Key2	Value2
{A,B}	6/8
{A,C}	0
{B,C}	6/8

Output of map2

Key2	Value2
{D,E}	6/8
{D,F}	4/8
{E,F}	6/8

Output of map3

Key2	Value2
{A,B}	5/8
{A,C}	4/8
{B,C}	1

Output of map4

Key2	Value2
{D,E}	0
{D,F}	3/8
{E,F}	5/8

Step 3: Output of Combine

Output of combine1

Key2	Value2
{A,B}	6/8 5/8
{A,C}	0 4/8
{B,C}	6/8 1

Output of combine2

Key2	Value2
{D,E}	6/8 0
{D,F}	4/8 3/8
{E,F}	6/8 5/8

Table 3: Number of records matrix removed redundant attributes values

U	B	C	E	F	G
1	1	1	1	1	1
2	1	2	1	2	2
3	1	1	2	1	1
4	1	2	2	2	2
5	2	1	1	1	1
6	2	2	1	2	2
7	2	1	2	1	2
8	2	2	2	2	2
9	11	2	1	1	1
10	1	1	1	1	1
11	2	1	2	1	2
12	2	1	2	2	2
13	2	2	1	1	2
14	2	2	1	2	2
15	2	2	2	1	2
16	1	2	2	2	2

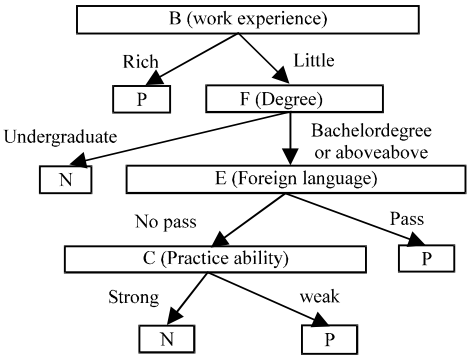

Fig. 3: Parallel decision tree based on MapReduce

Step 4: Output of reduce

Key2	Value2
{B,C}	14/8
{A,B}	11/8
{E,F}	11/8
{D,F}	7/8
{D,E}	6/8
{A,C}	4/8

Ultimately the results outputted by reduce will beretum to the user, allowing users to compare the size of the various properties and eventually come to the order of {B, C}>{A, B} = {E, F}>{D, F}>{DE}>{A, C}, {A, C} set of attributes dependence is 0 and A is the smallest one of the three attributes of the A, B, C and so it is a redundant data ,so it can be removed, with The rationale D can also be removed. The final draw of the sort is B>C, F>E, but the bad relationship between the four of them to determine, there are two ways to compare the size of the C and F, the first direct use of single attribute dependency, the best the situation is one time, is that F<C, but the worst case is three times, here is the kind of bad, so it would be a waste of time and the credibility of the single attribute dependence lower than attribute set dependence, so we re-write these four properties of the B, C, E, F to the number of values recorded matrix, as shown in Table 3.

Step 4: Continue to split the remaining attributes

At this time the number of attributes are relatively small, so using of MapReduce splits the training set only take horizontal split, divided into two sections 1-8 and 9-16, the final calculation attributes set dependence of {B, C, E} {B, C,F} {B, E, F} {C, E, F} are 7/8, 7/8, 1,5/8, drawn B>F, E>C and then combined with the above conclusions B>C, F>E can be derived B>F>E>C. the eventual establishment of the decision tree shown in Fig. 3:

The decision tree was structured as shown in Fig. 3, you can come to a series of rules, each path is a rule, it is so simple and clear and viewing from the algorithm, the time complexity is O (n)/p, where p is the number of the processor. Its time complexity is less than that of the serial's Cache data in memory will be written to disk, so it will not take up too much memory.

CONCLUSION

Attribute set dependence, combined with the traditional serial algorithm for the training of small-scale data sets can quickly constructing a decision tree, but for the large-scale data, construction of decision tree should use parallel programming model, abstraction of MapReduce parallel programming is high , core operating are two functions: Map and reduce, Map operation and reduce operations are highly parallel run, in order to achieve data processing highly parallel, the dependence of the attribute se run in the change model can not only reduce the time complexity, but also ensure that the structure of the decision tree is more accurate and reliable. The algorithm plays a role for increasing the efficiency of other algorithms.

ACKNOWLEDGMENTS

Supported by Natural Science Fund in JiangSu (BK2009535) and Jiangsu Province ordinary university innovative research project (CXZZ13_0691).

REFRENCES

Hu, C.J. and X.W. Wang, 2008. Research of parallel programming model based on Multi-core cluster system. Comput. Technol. Dev., 4: 70-73.

Li, C.H., X.F. Zhang, H. Jin and W. Xiang, 2011. MapReduce: A new programming model for distributed parallel computing. Comput. Eng. Sci., 3: 129-135.

Cheng, M. and H.P. Chen, 2011. Weblog mining based on hadoop. Comput. Eng., 11: 37-39.

Wang, H.C., D.J. Zhu, X.N. Cao and J.P. Fan, 2009. Research on hybrid parallel programming model based on SMP cluster. Comput. Eng., 3: 271-273.

Xiong, H., Z. Wang, Y. Liu and M. Yang, 2008. Constructing parallel cluster based on PVM in linux environment. Comput. Dev. Appl., 2: 52-54.

Yang, G., 2010. HADOOP of data mining research. Chongqing University, Chongqing, pp: 45-46.

Zhu, S.K. and J.Y. Wan, 2009. Parallel genetic algorithm skeleton research and realization. Comput. Eng. Des., 20: 4588-4591.

Zhu, M., J.Y. Wan and M.W. Wang, 2011. Parallel decision tree classification algorithm based on MR and realized. Guangxi Normal Univ. Nat. Sci., 29: 82-84.