

# A Classification Algorithm of CART Decision Tree based on MapReduce Attribute Weights

Fubao Zhu, Mengmeng Tang, Lijie Xie, Haodong Zhu \*

*School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, Henan, 450002, China*

---

## Abstract

A CART decision tree algorithm based on attribute weight is proposed in this paper because of the present problems of complex classification, poor accuracy, low efficiency, and severe memory consumption of CART decision. What is more, the algorithm is combined with the parallel computing model of MapReduce. Theory of attribute weights is used in the algorithm. A decision tree is built through the sum of weights, which is decided by the degree that the attributes affect a decision. Thus the accuracy of classification through decision tree is improved. Parallel sorting algorithms of CART decision tree for massive data is implemented through the MapReduce programming technology of cloud computing. All the results of theoretical analysis and experimental comparison show that it is very important to mark attributes by weights through MapReduce. Furthermore, the accuracy of the classification of large sample data sets is improved significantly, classification efficiency of decision tree is improved and the trained time is also significantly reduced.

*Keywords:* MapReduce; attribute weights; decision tree algorithm; classification

(Submitted on July 24, 2017; Revised on September 25, 2017; Accepted on November 29, 2017)

© 2018 Totem Publisher, Inc. All rights reserved.

---

## 1. Introduction

In data mining, decision tree is a technology that is frequently used in data analysis and forecasting. With the advent of the era of big data, the growth rate of the amount of data is accelerating [1]. How to deal with data within the effective time has become one of the most concerning problems. If the data can't be timely processed after it is generated, the data will lose its value, and data mining will not reflect its meaning. 5V characteristics of big data (proposed by IBM): Volume, Velocity, Variety, Value, and Veracity make the traditional data mining methods difficult to adapt it.

CART decision tree algorithm is one of the classical algorithms of decision. In order to overcome the short-comings of CART decision tree algorithms, some improved algorithms have been proposed. Song [3] uses discrete and disordered attributes as splitting nodes of the decision tree, and the split can produce multiple sub-nodes. Jaworski, Pietruczuk [4] proposed that take the best attributes of the available data samples as consideration nodes of split of the whole data flow. The best attributes can be calculated through Gauss's theorem according to the available data samples. Mahmood, Imran and Satuluri [5] think that the introduction of Gini index (RGI), a new heuristic function for dimensionality reduction, could reduce the dimension of data and improve the accuracy of data classification. Chen and Xia [6] use the decision tree algorithm under the categorical variables, and introduce an optimized splitting function that constructs binary trees using domain variables, through which it can extract and select the prediction criterion. Some scholars have put forward the method of using the Fayyad Boundary Points Decision Principle to improve the method of dividing the threshold thorough which CART decision trees select continuous attributes [7]. Therefore, researchers try to parallelize the traditional attribute reduction algorithms to improve their efficiency on massive data. The main idea of these improved algorithms is to reduce the dimension of the data, simplify the data set to be processed, and use the function calculation to select the optimal segmentation point. In the age of big data, the amount of data is large and data types are complex. Accuracy of CART

---

\* Corresponding author.

E-mail address: [zhuhaodong80@163.com](mailto:zhuhaodong80@163.com)

decision tree algorithm is poor when solving complex decision-making problems. In addition, memory and efficiency problems when CART decision tree algorithm processes big data have yet to be resolved. So, the difficulty of using the simplified data dimension and the optimal segmentation point to realize the classification of a large number of data has been increasing.

In recent years, with the rapid development of computer networks and data storage technology, the amount of information data has been growing in a geometric progression [2]. To process the massive data, a single computer running time will be very long using the current data mining serial algorithm, which makes it difficult to meet the needs of data processing. Because of the limited computing resources of a single computer, for a large data set, the entire data set can't be loaded into the memory at once, which increases pressure to the whole calculation process [3]. Even the serial decision tree algorithm can no longer generate the correct decision rules.

The MapReduce concept proposed by Google is currently a popular distributed computing framework [4]. Its computing model provides a new way for data processing. MapReduce has been successfully applied to other algorithms in data mining. However, there are few studies on improving the performance of CART using parallel computing weights. Parallel distributed computation is mainly used to improve the selection efficiency of the optimal points, and the concept of weights is introduced to simplify the selection of the split points. An improved algorithm is proposed in this paper, where the algorithm attribute weights are combined with CART decision trees in the MapReduce. In the algorithm, the input data files are divided into small files to be processed on the nodes, calculating attribute weights of the sample, selecting the attribute with the largest weight as the split point, and then the decision tree classification can be carried out to realize the parallel computation of the decision tree, which greatly reduces the use of the memory space and improves the efficiency of the classification. Experimental results show that the algorithm proposed in this paper can effectively realize the classification of large-scale data sets. The rest of this article is organized as follows: in Section 2, the MapReduce framework and the CART decision tree classification algorithm are reviewed; In Section 3, the parallelism of the weight attribute calculation process is proposed, and the attribute weights is used to select the optimal point to construct the CART decision tree; In Section 4, experimental results are introduced to verify the validity of the proposed CART classification improvement algorithm based on MapReduce attribute weights. Finally, in Section 5, the work of the paper is summarized.

## 2. Related concepts

### 2.1. MapReduce model

As early as 2004, MapReduce was first proposed by Google's Jeffery Dean and Sanjay Ghemawat. The name of MapReduce consists of two words, Map and Reduce, which are commonly used in functional programming. The core of MapReduce lies in the Map function and the Reduce function. Millions of processors are used for parallel computing, which consumes a very low cost of users. The program of users can be deployed on the cluster through the programming of Map function and Reduce function, and then the distributed functions of users' own can be realized without paying too much attention to the distributed parallel computing technology, which is transparent to programmers. Thus, difficulty of implementation is reduced. The idea of the function-based programming language is used for a reference in MapReduce. The implementation process of MapReduce is that a Map method is specified for mapping the passed in data to a new key value through the form of key-value [8], then the new key value will be used for subsequent processing. Reduce, as a constraint, combines values with the same key-value pair as a collection. Data processing mode of divide and rule is used in MapReduce at the beginning to deal with large-scale data [9]. In the model, data is distributed to various nodes for processing. MapReduce is divided into two stages of processing and analysis for data. When the data of each node is processed, data stored in the nearest local will be read. Then, corresponding merger and sorting will be done on the data, after which the data is sent to the Reduce for further processing. Through this process, the large-scale data transmission required by the large data processing framework is avoided, thus saving time and improving processing efficiency.

The workflow for processing jobs submitted by the user through MapReduce is as follows: The submitted job is divided into many small pieces; simultaneously the input data is divided into a number of fixed-size modules distributed to each node [10-11]; then, the operation of the node is done in the local. When the operation of the node is complete, the output results need some sort of recombination and resorting. After the resorting, according to the different keys generated on the Map, the intermediate data should be reassigned for the next step, and then the Reduce task acts on one key at a time and combines all of the values associated with the key in some way to output. The work flow chart of MapReduce is shown in Figure 1.

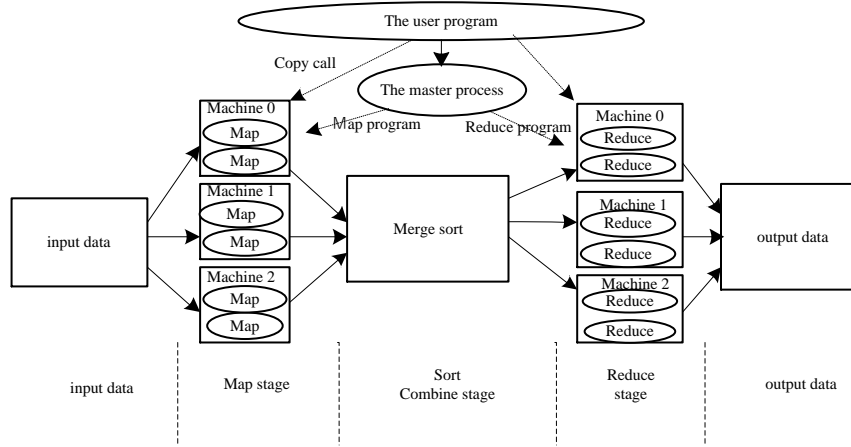


Figure 1. Working flow chart of MapReduce

## 2.2. Classification algorithm of CART decision tree

Decision tree is a classification rule method in the representation of decision tree, and the rule is reasoned from the sample set of which the data is unordered and irregular [12]. The top-down recursion method is used in decision tree. Comparison of attribute values is done on the internal nodes of the decision tree. Branches downing from a node is judged according to different attribute values, and conclusions can be derived in the leaf node of the decision tree [13-15]. So, a path from the root node to the leaf node corresponds to a rule, and the entire decision tree corresponds to a set of expression rules. Then, the generated decision tree can be defined as follows: A decision tree is a tree structure (either a binary or non-binary tree) with each non-leaf node representing a feature attribute, each branch representing the output of the feature attribute on a range of values, and each leaf node storing a category [16]. The process of decision-making with decision tree starts from the root node, testing the corresponding feature attribute in the item to be classified, then selecting the output branch according to its attribute value until the leaf node is reached, and then the category stored in the leaf node is used as the decision result.

The classification algorithm of CART decision tree was proposed by BreimanL, Friedman J H and Olshen R A in 1984 [17]. Its application is very extensive, as it is one of the important methods of decision tree classification. Feature selection, tree generation and pruning are important parts of CART. A partitioning technique using binary recursion is used in CART [18-20]. The conditional probability distribution of random variable Y is output under given input random variable X, and the current set is divided into two subsets, making it a sub-node of the decision tree.

Process of CART classification is the process that the training set is divided into smaller and smaller subsets [21-23]. The ideal result is that there is the same tag to leaf node samples of generate tree. The selection criterion of CART nodes is to make the impurity of the nodes as small as possible. The smallest Gini coefficient of each attribute is used as the standard for selecting test attributes in CART. Supposing there are k classes, the probability that a sample point belongs to the k class is  $P_k$ , then the Gini index of the probability distribution is:

$$Gini(T) = 1 - \sum_{k=1}^k P_k^2 \quad (1)$$

If the eigenvalue of sample set T is A, the set T can be divided into two parts of T<sub>1</sub> and T<sub>2</sub> by a taken out according to the eigenvalue [24]. Then the Gini index of T is defined as(2):

$$Gini(T, A) = \frac{|T_1|}{|T|} Gini(T_1) + \frac{|T_2|}{|T|} Gini(T_2) \quad (2)$$

The larger the Gini index calculated from  $Gini(T, A)$ , the greater the uncertainty of the sample set will be. Precision of decision tree generated through CART algorithm is high, while the complexity is not very high.

## 3. CART decision tree algorithm based on MapReduce attribute weights

**The Thinking of algorithm.** Generation of decision tree is an iterative process [25]. If the traditional serial algorithm is

used to achieve the process, a small amount of data also needs to spend a lot of resources, not to mention very large-scale data sets. This problem can be solved with the parallel programming method. CART decision tree is also generated through repeated iterations. In the case of large amounts of data, it is difficult to meet the needs of data mining with a single node. The most time-consuming process of constructing CART calculating the GINI coefficients [8]. For this problem, the algorithm can be applied to the MapReduce, because the computation between the attributes can be performed in parallel [26-27]. As a measure of uncertainty, the GINI index is the same as the information entropy. In order to simplify the model and not completely lose the entropy model, the GINI coefficient, instead of the information gain ratio, is used in the CART classification tree algorithm. The GINI coefficient represents the impurity of the model. The smaller the GINI coefficient is, the lower the purity and the better the feature will be. This is opposite to the information gain ratio. In deciding which attribute is currently the best classification attribute, the general practice is to exhaust all the existing attributes, and the fragmentation of each attribute is quantified to calculate the best split. In order to choose the split point more easily and improve the "purity" of the collection to be sorted, the concept of weight is introduced to the improved algorithm, and the weight of attribute is used as the priority condition for its selection. CART decision tree algorithm based on MapReduce attribute weights is a decision tree algorithm based on attribute weights. The influence level of different attributes on decision results is also different, according to which weights are obtained from the training data. The attribute with the largest weight is chosen as the splitting attribute. The larger the weight value is, the closer it is to the root node. Then calculate Gini coefficient of the features, and select the features with the smallest Gini index and its corresponding cut-off point, which can be seen as the concrete realization of algorithm of optimal feature and the optimal cut point. According to the previous theory, the input file is divided into file blocks. MapReduce provides multiple Map tasks; each task is one or more file blocks [28]. The main task of the map process is to generate key value pairs (attribute name  $k$ , attribute value). The Reducer organizes the aggregated data, counting the number of all attribute values of each attribute, calculating GINI coefficient of an individual attribute, determining the optimal cut point of a single attribute and calculating the weight value of a single attribute.

Suppose the occurrence number of attribute  $a_K$  ( $K = 1, 2, \dots, r$ ) is  $M_K$  and the total number is  $N$ . Then the frequency of  $a_K$  can be given as (3):

$$Ta_k = \frac{M_K}{N} \quad (3)$$

The weight of the attribute  $a_K$  can be Calculated as (4):

$$Wa_k = \frac{Ta_k}{\sum_{k=1}^r Ta_k} \quad (4)$$

The sum of the weights of all attributes satisfies the equation given below (5):

$$\sum_{k=1}^r Wa_k = 1 \quad (5)$$

According to the definition of weight theory, the specific implementation of CART decision tree algorithm based on MapReduce attribute weight is described in detail as follows:

Train sample set  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$

Train attribute set  $A = \{a_1, a_2, \dots, a_n\}$

TreeGenerate( $D, A$ ):

    Calculate the corresponding weight of all attributes, and then select the node with the largest attribute weight as the split point, marked as node

    If  $D$  all belongs to the same category  $C$ :

        Mark node as a class  $C$  leaf node;

        Return;

    End if

    If attribute set  $A$  is empty or all attribute values of  $D$  are the same:

        Mark the node as a leaf node, and its class is marked as the class with the largest number of samples in  $D$ ;

        Return;

    End if

        If the recursion depth of TreeGenerate( $D_v, A \setminus \{a^*\}$ ) reaching the customer's needs for analysis and application

        Return;

```

Else if Calculate the weight of all attribute weights in class A to be sorted, then select the best partition
attribute a*
for a in a*:
    Generate a branch for node, so that Dv represents a subset of the samples in D of which the a * attribute
    value is a;
    If Dv is empty then
        Mark the branch node as a leaf node, and its class is marked as the class with the largest number of
        samples in D;
        Return;
    Else
        Recursion of TreeGenerate(Dv, A \ {a*}) continues
    End if
End for

```

Flow chart of the algorithm is shown in Figure 2.

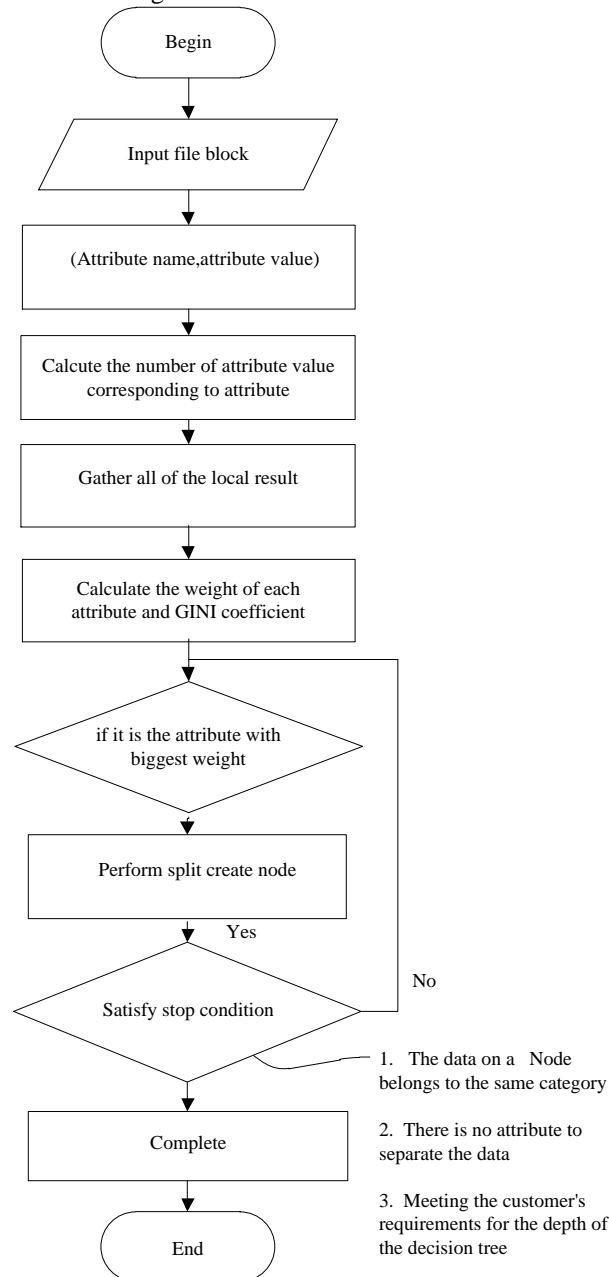


Figure 2. Flow chart of the algorithm

#### 4. Simulation experiments and analysis

In order to verify the effectiveness of the algorithm, an experiment is done. Since a part of the core of Hadoop's framework is MapReduce, which provides computations for massive amounts of data, the Hadoop platform is chosen as the operating environment for the experiment. The Hadoop 2.2.0 classic version is used. Six PCs with win7 operating system are used as Hardware devices and Intel i7 processor is used as the main device node. The memory size of the processor is 8GB. Intel i7 processor is also used as slave devices, while the memory size is 4GB, and the frequency is 3.6GHz. A customer database of a real estate company is used as the main data source of test data. Information of visiting customers, telephone counseling and purchase customers of nearly three years is included in the database. The information of the original data of a real estate company can be seen in Table 1.

Table 1. Information of original data of a real estate company

Data set	Number of objects	Number of samples	Data set	Number of objects	Number of samples
Building A	customer	19868	Building C	customer	43685
Building A	house	10593	Building C	house	49856
Building A	others	9307	Building C	others	29652
Building B	customer	39608	Building D	customer	52584
Building B	house	35685	Building D	house	56253
Building B	others	22569	Building D	others	45685

The result of attribute weight of the information calculated with the formula of weight value can be seen in Table 2.

Table 2. Calculation results of attribute weight

Customer	Weight	House	Weight	Other	Weight
age	0.162	Housing area	0.045	Propaganda media	0.028
region	0.013	purpose	0.009	Surroundings	0.078
job	0.012	Building type	0.027	location	0.151
Loan situation	0.011	house type	0.052		
Payment method	0.039	storey	0.064		
Total budget	0.146	price	0.165		

In the first group experiment, building A with the smallest sample size is selected as the data model. Two algorithms are used to conduct the experiment. The compare results of the two algorithms to the data of building A is shown in Figure 3 and Figure 4. Figure 3 shows a time comparison of the first sets. Figure 4 shows an accuracy comparison of the first sets.

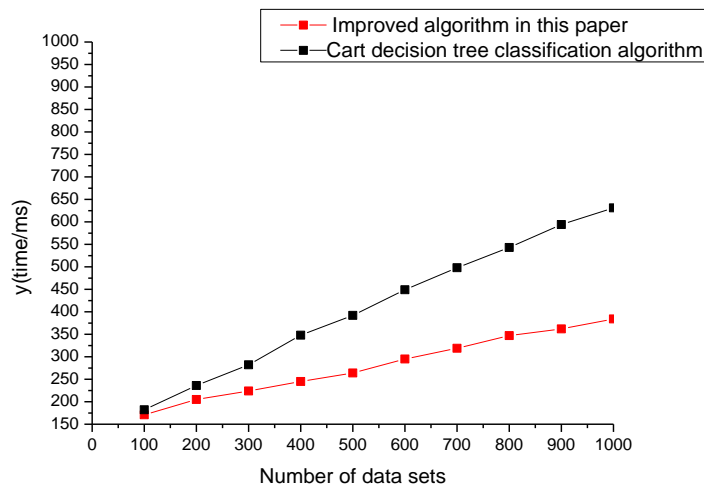


Figure 3. Time comparison of the first sets

In order to simulate the larger data, all the data of building A, B, C, D is selected in the second group of experiments. The real data sets are reused to make more data comparisons. The compare results of the algorithm to large data in the experiments are shown in Figure 5 and Figure 6. Figure 5 shows a time comparison of the second sets. Fig.6 shows an accuracy comparison of the second sets.

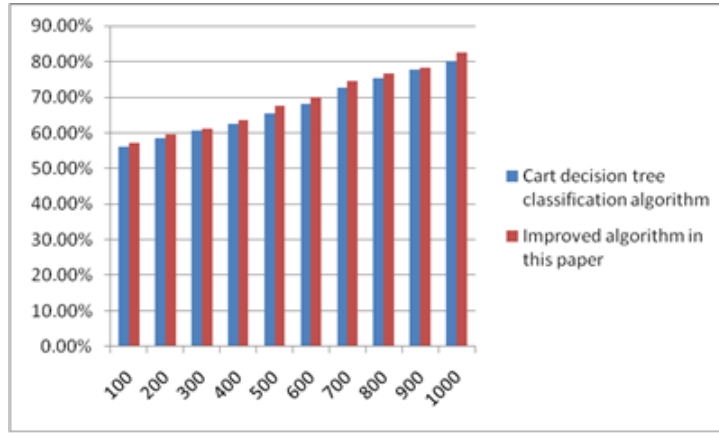


Figure 4. Accuracy comparison of the first sets

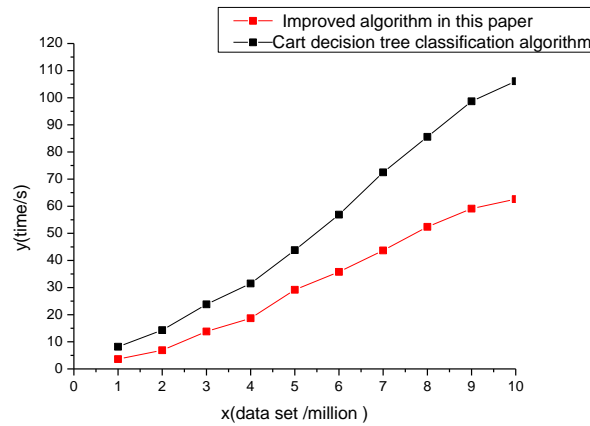


Figure 5. Time comparison of the second sets

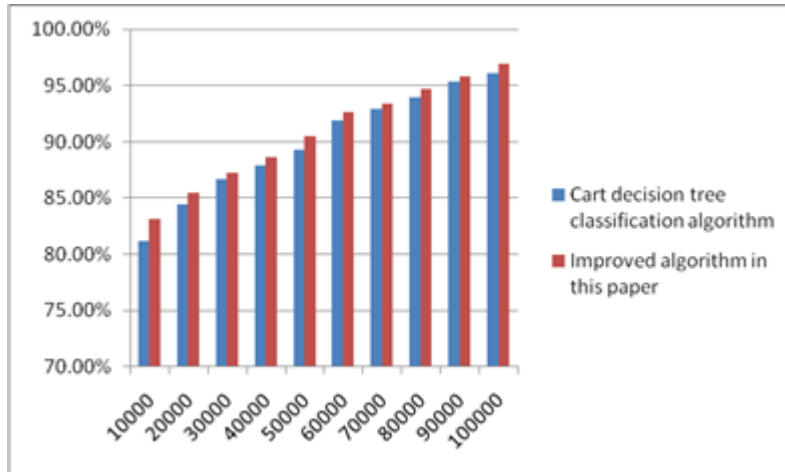


Figure 6. Accuracy comparison of the second sets

It can be seen from Fig 3 and Fig 4 that the execution time of this algorithm is the same as that of traditional algorithm to the data set of building A. That is mainly because the algorithm in the implementation of the process of calculating the property weights also needs to spend part of the time. For a small data set, though splitting point of the attributes can be quickly determined, the time reduction is not obvious in the overall effect because the data set is small. But, accuracy of the algorithm is higher. It can be seen from Fig 5 and Fig 6 that the algorithm not only outperforms the traditional algorithm in execution time, but also outperforms the CART decision tree algorithm in algorithm accuracy for the simulated large data stream dataset. The results show that running time of CART decision tree algorithm based on MapReduce attribute weights is shorter than the running time of traditional CART decision tree algorithm, and it is superior to traditional algorithm in the

efficiency of the algorithm, and accuracy of the algorithm is higher. The main difference between this algorithm and the traditional CART decision tree algorithm lies in follows: In the parallel computing based on MapReduce proposed in this paper, the weights of different attributes are computed and assigned to different work nodes under the premise that the weights are calculated accurately. After the calculation and integrating the results, the GINI coefficients of the splitting properties are calculated in parallel in the next step to reduce the computation of the algorithm in single node calculation quantity, thus improving the efficiency of algorithm implementation.

## 5. Conclusions

For the problem that the improved algorithm for simple data sets cannot effectively deal with large data sets, an improved CART decision tree algorithm based on MapReduce attribute weight is proposed in this paper. The characteristic that the attribute weight is proportional to its influence degree is used to select the attribute with the largest weight in the information system as the current split attribute. The concept of weight is used to support the branching process, which improves the efficiency of generating decision tree. Through the example, it is verified that the rules generated by the decision tree can not only correctly classify the sample data, but also reduce the complexity of the spanning tree. After comparing the value of weights, the attributes with the largest weights are selected as the splitting attributes, which improves the efficiency of the spanning tree. The experimental results also show that decision tree created by weight is more accurate in classification than that of the traditional algorithm.

In addition, the accuracy of the improved algorithm is slightly improved, but still low, which is the shortcoming of the CART algorithm. Further research with other classification algorithms, such as SVM algorithm will be done to improve the accuracy of sample set classification. At the same time, the rough set model will be introduced to study the parallel optimization of other simplified algorithms.

## Acknowledgements

The authors would like to thank the editors and the anonymous reviewers for their helpful comments and suggestions, which have improved the presentation. This work was supported in part by the Science and Technology Plan Projects of Henan Province of China under grant No. 162102210059, No. 152102210357 and No. 152102210149, the Youth Backbone Teachers Funding Planning Project of Colleges and Universities in Henan Province of China under grant No. 2014GGJS-084, the Key Science Research Project of Colleges and Universities in Henan Province of China under grant No. 16A520030, the Youth Backbone Teachers Training Targets Funded Project of Zhengzhou University of Light Industry of Henan Province of China under grant No. XGGJS02, the Ph.D. Research Funded Project of Zhengzhou University of Light Industry of Henan Province of China under grant No. 2010BSJJ038 and No. 2014BSJJ080, and the National Science Foundation of China under grant No. 31640013, No. 61602425 and No. 81501548.

## References

1. A. Bar-Hen, S. Gey, J. M. Poggi, "Influence Measures for CART Classification Trees," *Journal of Classification*, Vol. 32, pp. 1-25, 2015.
2. R. C. Barros, M. P. Basgalupp, "Towards the automatic design of decision tree induction algorithms," *Springerbriefs in Computer Science*, Vol. 13, pp. 567-574, 2015.
3. A. Bechini, F. Marcelloni, A. Segatori, "A MapReduce solution for associative classification of big data," *Information Sciences An International Journal*, Vol. 332, pp. 33-55, 2016.
4. Y. Ben-Haim, E. Tom-Tov, "A Streaming Parallel Decision Tree Algorithm," *Journal of Machine Learning Research*, Vol. 11, pp. 849-872, 2010.
5. B. Chandra, R. Kothari, P. Paul, "A new node splitting measure for decision tree construction," *Pattern Recognition*, Vol. 43, pp. 2725-2731, 2010.
6. L. Chasmer, C. Hopkinson, T. Veness, et al, "A decision-tree classification for low-lying complex land cover types within the zone of discontinuous permafrost," *Remote Sensing of Environment*, Vol. 143, pp. 73-84, 2014.
7. H. L. Chen, D. X. Xia, "Applied Research on Data Mining Based on CART Decision Tree Algorithm," *Coal Technology*, Vol. 30, pp. 164-166, 2011.
8. M. R. Hassan, R. Kotagiri, "A new approach to enhance the performance of decision tree for classifying gene expression data," *Bmc Proceedings*, Vol. 7, pp. 1-8, 2013.
9. K. S. Hong, P. L. Ooi, C. K. Ye, "Multivariate alternating decision trees," *Pattern Recognition*, Vol. 50, pp. 195-209, 2016.
10. M. Jovanović, B. Delibašić, M. Vukićević, "Evolutionary approach for automated component-based decision tree algorithm design," *Intelligent Data Analysis*, Vol. 18, pp. 63-77, 2014.
11. S. Y. Kim, A. Upneja, "Predicting restaurant financial distress using decision tree and AdaBoosted decision tree models," *Economic Modelling*, Vol. 36, pp. 354-362, 2014.



12. M. Kumar, S. K. Rath, "Classification of microarray using MapReduce based proximal support vector machine classifier," *Knowledge-Based Systems*, Vol. 89, pp. 584-602, 2015.
13. L. Li, "A Bayes Classifier-Based OVFD Algorithm for Massive Stream Data Mining on Big Data Platform." Conference on Complex, Intelligent, and Software Intensive Systems. Springer, Cham, pp. 537-546, 2017.
14. J. Liu, Y. H. Li, M. M. Yang, "Decision Tree Algorithm Based on MapReduce in Telecommunications Churn," *Computer Knowledge & Technology*, Vol. 30, pp. 6710-6713+6716, 2013.
15. L. Ma, S. Destercke, Y. Wang, "Online active learning of decision trees with evidential data," *Pattern Recognition*, Vol. 52, pp. 33-45, 2016.
16. A. M. Mahmood, M. Imran, N. Satuluri, "An Improved CART Decision Tree for Datasets with Irrelevant Feature," *Swarm, Evolutionary, and Memetic Computing*, pp. 539-549, 2011.
17. D. Mudali, L. K. Teune, R. J. Renken, "Classification of Parkinsonian syndromes from FDG-PET brain data using decision trees with SSM/PCA features," *Computational & Mathematical Methods in Medicine*, Vol. 2015, pp. 1-10, 2015.
18. V. Purdilă, S. G. Pentiu, "MR-Tree - A Scalable MapReduce Algorithm for Building Decision Trees," *Journal of Applied Computer Science & Mathematics*, Vol. 8, pp. 16-19, 2014.
19. J. Qian, D. Miao, Z. Zhang, "Parallel attribute reduction algorithms using MapReduce," *Information Sciences*, Vol. 279, pp. 671-690, 2014.
20. L. U. Qiu, X. H. Cheng, "Parallelization of decision tree algorithm based on MapReduce," *Journal of Computer Applications*, Vol. 32, pp. 2463-2462+2469, 2012.
21. L. Rutkowski, M. Jaworski, L. Pietruczuk, "The CART decision tree for mining data streams," *Information Sciences*, Vol. 266, pp. 1-15, 2014.
22. F. Saqib, A. Dutta, J. Plusquellic, "Pipelined Decision Tree Classification Accelerator Implementation in FPGA (DT-CAIF)," *IEEE Transactions on Computers*, Vol. 64, pp. 280-285, 2015.
23. G. L. Song, Z. X. Hao, "An Improved Algorithm Based on CART Decision," *Journal of Harbin University of Science and Technology*, Vol. 14, pp. 17-20, 2009.
24. D. Tapiador, W. O'Mullane, A. G. A. Brown, "A framework for building hypercubes using MapReduce," *Computer Physics Communications*, Vol. 185, pp. 1429-1438, 2014.
25. I. Triguero, D. Peralta, J. Bacardit, "MRPR: A MapReduce solution for prototype reduction in big data classification," *Neurocomputing*, Vol. 150, pp. 331-345, 2015.
26. W. Wang, K. Zhu, L. Ying, "Map Task Scheduling in MapReduce With Data Locality: Throughput and Heavy-Traffic Optimality," *IEEE/ACM Transactions on Networking*, Vol. 24, pp. 190-203, 2016.
27. M. Zeinalkhani, M. Eftekhari, "Comparing Different Stopping Criteria for Fuzzy Decision Tree Induction through IDFID3," *Iranian Journal of Fuzzy Systems*, Vol. 11, pp. 27-48, 2014.
28. L. Zhang, Q. Ning, "Two improvements on CART decision tree and its application," *Computer Engineering and Design*, Vol. 36, pp. 1209-1213, 2015.