Distributed Storage Big Data Parallel Clustering Scheduling Algorithm Based on Decision Tree

Pan Jinfeng, Hu Xiaoqin

Quanzhou University of Information Engineering, Quanzhou, Fujian, 362000, China
panjinf2002@163.com

Abstract: *To optimize the parallel clustering scheduling effect of distributed storage big data, this study proposes the design of a decision tree based parallel clustering scheduling algorithm for distributed storage big data. This method first constructs a decision tree model to achieve distributed storage data classification, and then implements parallel design of the decision tree model based on the Storm platform to improve data classification speed. Finally, based on this, a distributed storage data clustering scheduling design is implemented using an improved BPSO algorithm. The experimental results show that the proposed method has a shorter scheduling time and is superior to traditional methods, with better application results.*

Keywords: *Decision tree; Distributed storage big data; Parallel clustering scheduling*

1. Introduction

Big data clustering is a commonly used data analysis method that groups datasets based on their similarities and identifies patterns and correlations within them. However, traditional algorithms often cannot meet the needs of processing large-scale data [1]. Therefore, developing efficient parallel big data clustering algorithms has become an important research direction. Clustering big data in a distributed storage environment requires addressing challenges such as uneven data distribution, communication latency, and load balancing. Therefore, studying parallel clustering scheduling algorithms for distributed storage of big data is of great significance. By designing parallel clustering algorithms and scheduling strategies, the performance of big data clustering can be improved. Through reasonable task allocation and data partitioning, communication overhead can be reduced, computational efficiency can be improved, and load balancing and parallelization processing can be achieved. In this context, this study introduces decision tree theory and conducts research on parallel clustering scheduling algorithms for distributed storage of big data.

2. Build a distributed storage data classification decision tree model

To achieve parallel clustering scheduling of distributed storage big data, a decision tree model is constructed to achieve distributed storage data classification. The steps are as follows: first, preprocess the original data by cleaning, removing noise, and processing missing values [2]; Then, based on the correlation and importance of features, a decision tree model is constructed using the information gain ratio as the feature branching criterion. The expression of the information gain ratio is as follows:

$$Gain_r = \frac{Gain(T,f)}{IV(f)} \tag{1}$$

1) In the formula, $Gain(T,f)$ represents information gain; T represents the dataset; f represents the discrete characteristics of the data, and $IV(f)$represents the inherent attributes of feature f [3]. Finally, using the selected features and judgment conditions, a decision tree model is recursively constructed, and the dataset is divided by traversing different features and judgment conditions.

3. Decision Tree Parallelization Design Based on Storm

After completing the construction of the decision tree model, Storm was used to implement the parallelization design of the decision tree. Storm is a computing platform that provides support for Spout and Bolt programming interfaces. The Spout component is used for inputting data streams, while the Bolt component is used to process data logic. Based on the above, a parallel computing architecture design for decision trees is implemented, as shown in Figure 1.

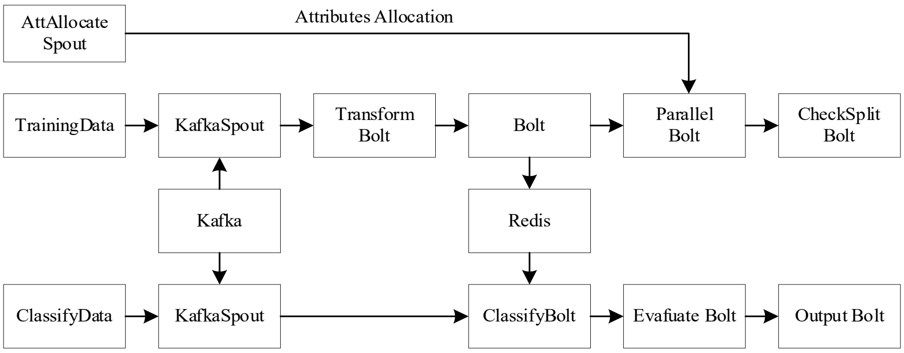

Figure 1: Decision Tree Parallel Computing Architecture Design

As shown in Figure 1, the function of AttAllocate Spout is to read the attribute set file, which records the number of attributes and their corresponding discrete attribute values. The attributes are allocated through the set parallelism, and the attribute class Attribute is passed to ParallelBolt. KafkaSpout can read the data in the training dataset one by one through a file and pass the data in tuple form to TransformBolt. The tuples received by TransformBolt are string type data one by one and cannot be directly handed over to the tree building module for training. Therefore, it will convert the passed data into Instance format and pass it to Bolt for training, achieving the update of statistical values of leaf nodes [4]. When the number of training samples counted by a leaf node is nl% nmin=0, it is necessary to calculate the information gain of all attributes on that leaf node. The parallelization calculation of attribute information gain is entrusted to the ParallelBolt below for completion. The content passed by Bolt to ParallelBolt is the leaf nodes that currently have reached the minimum number of split samples and need to be split. The node and other related statistics are sent to all threads of ParallelBolt using the message grouping method of AllGrouping(). In summary, complete the parallel computing design of the decision tree algorithm.

4. Design of Distributed Storage Data Clustering Scheduling

After completing the distributed storage data classification decision tree model and parallel computing design, the efficiency of distributed storage data partitioning can be improved. Next, using it as a data sample, the improved BPSO algorithm is used to implement the distributed storage data clustering scheduling design. The assignment relationship between Design Task Set $T=(t_1,t_2,\cdots,t_n)$and Dataset$V=(v_1,v_2,\cdots,v_m)$is represented by the following equation (2):

$$X=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{12} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \tag{2}$$

In the formula, x_{mn} represents the allocation relationship between data and scheduling tasks. If scheduling task t_n is executed on data x_m, then $x_{mn}=1$; Otherwise, if $x_{mn}=0$, there is $\sum_{i=1}^{m} x_{ij}=1$, where $i=\{1,2,\cdots,m\}$, $j=\{1,2,\cdots,n\}$ [5,6]. The allocation relationship matrix between corresponding tasks and data is as follows:

$$E=\begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{12} & e_{22} & \cdots & e_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ e_{m1} & e_{m2} & \cdots & e_{mn} \end{bmatrix} \tag{3}$$

In the formula, e_{mn} represents the time when Data x_m runs Task t_n. Let b_j be the time when task t_j started running, and c_j be the time when task t_j completed running, with the presence of $c_j = b_j + E_j$. Based on this, set the data clustering scheduling objective function to:

$$\min F = c_{j_{\max}} \tag{4}$$

In the formula, c_j max represents the total time required for task scheduling. Next, based on the improved BPSO algorithm, the objective function is solved, and the process is shown in Figure 2:

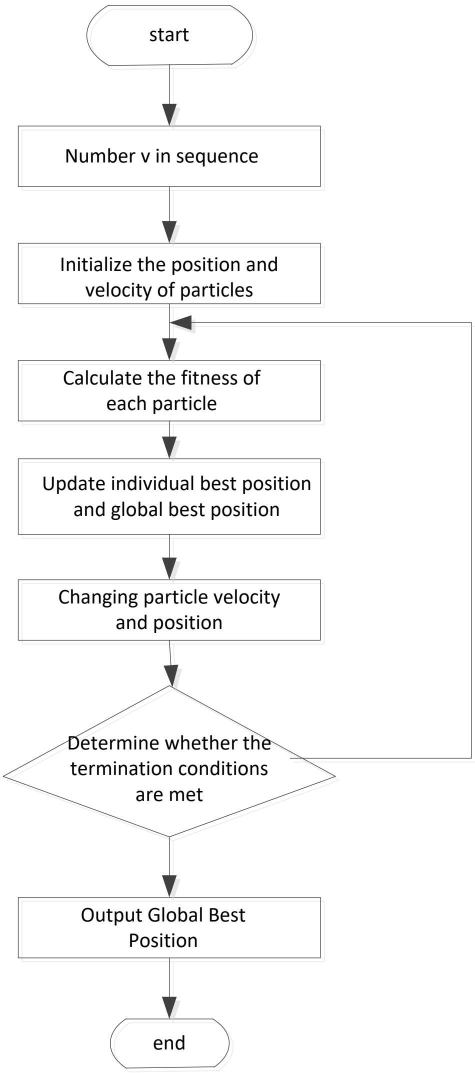
start
Number v in sequence
Initialize the position and velocity of particles
Calculate the fitness of each particle
Update individual best position and global best position
Changing particle velocity and position
Determine whether the termination conditions are met
Output Global Best Position
end

Figure 2: Flow Chart of Improved BPSO Algorithm Solution

Through the above, complete the design of distributed storage data clustering scheduling.

5. Experiments and Analysis

To verify the progressiveness of the proposed algorithm, seven standardized classification datasets are selected and saved on the cluster in ARFF format for algorithm testing. The basic information of the 7 datasets used is shown in Table 1.

Table 1: Basic Information of the Dataset

Number	Data Set	Samp672les	Attributes	Catgory
1	Car Evaluation	1572	7	16
2	Breast_cancer	678	3	4
3	Abalone	4371	1	6
4	Cmc	1782	2	5
5	Statlog	1382	2	9
6	Yeast	1462	3	7
7	Nursery	7839	3	8

Based on the above, complete data analysis on a Windows 10 laptop operating system. Selecting traditional methods as comparative methods for data clustering scheduling, the accuracy of the two methods' algorithms for data classification is first evaluated, and the comparison results are shown in Table 2.

Table 2: Comparison of Data Classification Accuracy

Data Set	Classification accuracy of the proposed algorithm/%	Traditional algorithm classification accuracy/%
Car Evaluation	88.13	75.31
Breast_cancer	89.76	74.89
Abalone	91.32	77.63
Cmc	94.16	73.86
Statlog	93.21	78.91
Yeast	92.11	76.75
Nursery	91.26	77.92

As shown in Table 2 above, the classification accuracy of the proposed method is higher, proving its better application performance. Next, compare the efficiency of two methods for data scheduling, and the comparison results are shown in Figure 3:

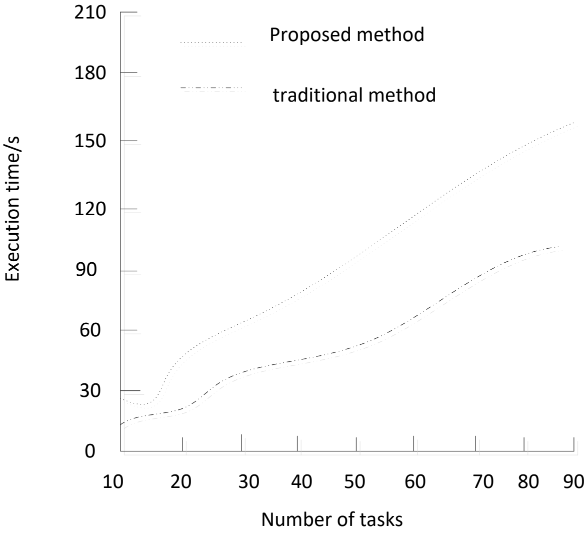

Figure 3: Comparison of Task Completion Times

As shown in Figure 3, as the number of tasks increases, the processing time also increases, with a linear trend. Compared to traditional methods, the proposed method consumes less time and the gap between the two becomes increasingly apparent as tasks increase. This is because traditional methods ignore the resources required by the task itself when scheduling data clustering, so the proposed method has a good application effect.

6. Conclusion

This study designed a decision tree based distributed storage big data parallel clustering scheduling algorithm to achieve optimization of distributed storage big data parallel clustering scheduling. The experimental results show that the proposed method has a shorter scheduling time in a distributed environment and can effectively improve the efficiency of parallel clustering scheduling for distributed storage big data. It provides an effective solution for the problem of parallel clustering scheduling for distributed storage big data. In future research, methods based on other optimization algorithms or combined with machine learning technology will be further explored to further improve the accuracy of parallel clustering scheduling for distributed storage of big data, in order to meet the growing demand for big data processing.

References

[1] Han Litao. Design of Parallel K-means Clustering Algorithm Based on Cloud Computing[J]. Information and Computers (Theoretical Edition), 2023 ,35(09):93-95.

[2] Mao Yimin, Gan Dejin, Liao Lefa, et al. Parallel division clustering algorithm based on Spark framework and ASPSO[J]. Journal on Communications,2022,43(03):148-163.

[3] Wang Yuxian. Research on big data parallel search clustering algorithm based on Cloud Computing [J]. Automation & Instrumentation,2021,(10):33-36.

[4] Liu Jiefang, Zhang Zhihui. Parallel clustering algorithm for big data[J]. Computer Engineering and Design,2021,42(08):2265-2270.

[5] Lin Xiaohong, Lu Xinghua, Ma Miantao, et al Data Parallel Clustering Mining Algorithm Based on Continuous Detail Feature Decomposition[J]. Computer Technology and Development, 2022, 32(04): 34-38.

[6] Zhao Chunxia, Zhao Yingying, Song Xuekun.Parallel Clustering Algorithm for Multi-source Heterogeneous Data Based on Frequent Itemsets[J]. Journal of University of Jinan(Science and Technology), 2022,36(04):440-443+451.