Optimization of K-means Algorithm Base on MapReduce

Xueyu Huang* and Shichao Cheng
School of Information Engineering, Jiangxi University of Science & Technology, Ganzhou 341000, China

*Corresponding author e-mail: huangxy@jxust.edu.cn

Abstract. In the context of big data, the K-means clustering algorithm is sensitive to the initial clustering center, the detection and removal of outliers are difficult, and the data is tilted. The KMEANS-BRMS (K-means algorithm based on range mean and sampling) algorithm based on range mean and sampling is proposed. First, for the initial data set, the "range mean value method" (Mean Range method, MRM) is proposed to obtain the initial cluster centers. In order to eliminate the influence of outliers, a range threshold is set above and below the range mean, and all points under this threshold constitute an initial center. The set of points to be selected, in which the maximum and minimum distance criteria are adopted to select the initial clustering center, which effectively avoids the problem of the sensitivity of the initial center point and the influence of outliers caused by the random selection of the initial clustering center. Next, the BSA (Based on pond sampling and first adaptation algorithm) strategy of adapting algorithm for the first time to deal with the data skew problem in the MapReduce stage, and improve the clustering efficiency. Finally, combined with the MapReduce framework model, the data cluster center is mining in parallel to generate the final clustering result. Experiments show that the clustering results of the KMEANS-BRMS algorithm are more stable and effective, and at the same time it can more effectively improve the efficiency of parallel computing in the big data environment.

Keywords: Big Data, K-Means Algorithm, Outliers, Extreme Errors, MapReduce

K-means algorithm as a traditional clustering algorithm has the characteristics of simple logic, easy to understand and strong practicability, so it is loved by people [1, 2]. The purpose of K-means algorithm is to put high similarity data in the same cluster and low similarity data in different clusters based on similarity. Because k-means algorithm can find potential distribution features from sample data, it is widely used in image recognition, data mining, anomaly detection, information retrieval and other fields.

With the rapid development of Internet technology and the arrival of big data era, traditional data has gradually developed into big data with 5V characteristics. 5V means volume, variety, velocity, accuracy and value. However, K-means algorithm is a serial algorithm with high time complexity. Most of them are applied to small-scale data, and the time complexity required for big data is huge, which cannot be met by most of the current hardware facilities. Therefore, under the background of
big data era, how to reduce the operation time of K-means algorithm and make it suitable for big data is a key problem [3].

In the era of big data, the scalability and efficiency of MapReduce have attracted people's attention, and it shows a strong ability in processing massive data. MapReduce is a parallel processing model developed by Google to process large-scale data. The traditional K-means algorithm combined with MapReduce model can change the algorithm from serial operation to parallel operation, which further reduces the time complexity of the algorithm. In reference, a divide and conquer K-means clustering method based on MapReduce is proposed [4]. In order to solve the problems of long execution time and poor clustering results, the algorithms adopts the idea of divide and conquer method, divides the data set into multiple modules and stores them in different machines to cluster independently. This method greatly reduces the clustering time and improves the clustering effect. However, due to the random selection of the initial clustering center and the interference of outliers [5], the clustering results are not stable. For this reason, reference proposed to use genetic algorithm to optimize the selection of the initial center [6], to solve the problem that the algorithm is sensitive to the initial clustering center. Ali F proposed a clustering algorithm based on the center of gravity [7]. The idea of the algorithm is to find clusters by the center of gravity, which is a point in the cluster. By finding the critical distance λ, the distance is used to determine the clusters with the least effective points and sharing the same threshold value. Then, the points with the lowest edge value are selected as the center of gravity of the cluster. Finally, the similarity of each point in the data set is determined by connectivity and cohesion [8-10]. This method can improve the poor clustering results caused by random initial clustering centers and irregular data sets.

To solve the above problems, this paper proposes KMEANS-BRMS (K-means algorithm based on range mean and sampling) algorithm based on range mean and sampling. The main works are as follows: (1) proposes the mean range method (MRM) to obtain the initial clustering center, which can avoid the instability of clustering results or local optimization caused by the random selection of initial clustering centers by traditional K-means algorithm, and reduce the interference of outliers; (2) proposes a BSA (based on pond sampling and first adaptation algorithm) strategy based on pond sampling and first adaptation algorithm to solve the data skew problem caused by uneven data distribution among nodes in the process of parallelization, so as to improve the overall cluster efficiency; (3) under the MapReduce framework, the improved k-means algorithm is run and the final clustering result is obtained.

1. Introduction of Related Concepts

1.1. Traditional K-Means Algorithm
K-means algorithm is a common clustering algorithm, which has a good clustering effect for small data sets. The core of K-means algorithm is to divide the data set of n samples into K clusters, so that the samples within each cluster have high similarity and the samples between clusters have low similarity. There are many standards to measure the similarity, which is also a research hotspot, but most of them use Euclidean distance as the measurement standard. The whole process of the algorithm is to select k samples randomly as the initial cluster center, then calculate the Euclidean distance from other samples in the set to the initial cluster center, divide the samples into corresponding clusters according to the distance, calculate the mean value of each cluster, and update the cluster center. Repeat the above process until the cluster center does not change or converge.

1.2. Mapreduce Computing Model
MapReduce is a simplified distributed programming model [12]. It is divided into two stages. Firstly, the input <key, value> is converted into the intermediate result <key', value'> in the map phase, and then the <K, list of V> is integrated in the reduce phase, and finally the <K', V> is generated.
1.3. Related Definitions

Let the set to be clustered be $D = (x_1, x_2, x_3 \ldots x_n)$ and $K$ initial clustering centers be expressed as $C_1, C_2 \ldots C_k$.

Definition 1 $\text{dis}(x_i, y_j)$ represents the Euclidean distance of $m$ dimension between object $x_i$ and $y_j$ in $D$.

$$\text{dis}(x_i, y_j) = \sqrt{\sum_{l=1}^{m} |x_{il} - y_{jl}|^2}$$ (1)

Where $m$ and $l$ denote the dimension of space.

Definition 2 Sum of squares of clustering errors:

$$E = \sum_{j=1}^{K} \sum_{x \in C_j} |x_l - C_j|^2$$ (2)

Definition 3 Maximum and minimum distance criterion [13]: Select any point in the set as the first cluster center $O_1$, object farthest from $O_1$ as the cluster center $O_2$. Then, when the remaining cluster centers $L$ ($L>2$) are selected, the minimum distance between the remaining sample points and the previous centers is put into a set $M = \{M_1, M_2 \ldots M_l\}$. In the algorithm, the object corresponding to the maximum value in the set $M$ is taken as the next cluster center, and the above process is repeated until $K$ cluster centers are obtained. The calculation formula is as follows:

$$\text{dis}_l = \max\{\min(\text{dis}_{i1}, \text{dis}_{i2} \ldots)(l, i = 1,2\ldots)\} (3)$$

Definition 4 First adaptation algorithm [14]: starting from the first node with remaining space, the intermediate data is allocated to the node that can first meet the requirements.

Definition 5 Pond sampling algorithm: Take the first $n$ numbers from the set $V$ and fill in the set $R$ as the set of random numbers, and then traverse $V[n]$ to generate the random number $M$, ranging from 0 to $n + i - 1$. If $M < n$, then:

$$V[m] = V[n + i - 1]$$ (4)

Where $i$ is the iteration number.

2. KMEANS-BRMS Algorithm

2.1. Algorithm Idea

KMEANS-BRMS mainly consists of the following three stages: the selection of the initial cluster center, the data partition in the intermediate stage and the final parallel clustering. (1) In the selection stage of initial cluster center: the range mean method is proposed to obtain the initial cluster center, which can avoid the interference of outliers similar to Keans++ algorithm [15]. At the same time, the initial cluster center selected by combining the maximum and minimum distance criterion can effectively avoid the sensitive problem of random initial center; (2) in the middle stage of data partitioning, a BSA strategy based on pond sampling and first fit algorithm is proposed to predict the distribution characteristics of the advanced data pool in the Map stage, and then the prediction data of the sampled data are partitioned according to the first fit algorithm, and all data partitions in the Map stage are scheduled according to the predicted partition scheme; (3) Parallel clustering stage: in MapReduce computing model, parallel mining cluster center, get the final clustering results.

2.2. Initial Cluster Center Acquisition

The traditional K-means algorithm mainly uses random selection to get the initial clustering center point. This method will have a great impact on the final clustering results, and the clustering effect is not good. Therefore, this paper proposes a method based on range mean to obtain the initial cluster center, which can avoid the problems of poor clustering results and sensitive initial cluster center caused by random selection. The range mean method is as follows:
Firstly, the Euclidean distance range \( V = \{V_1, V_2 \ldots V_n\} \) and of the mean value of sum range \( \bar{V} \) of each object in the whole dataset \( D \) is calculated according to definition 1 \( V = \{V_1, V_2 \ldots V_n\} \).

Secondly, the object closest to \( V \) is selected as the first initial cluster center, then:

\[
\min C_x = \min\{|V_l - \bar{V}|\} (l = 1, 2\ldots n) \tag{5}
\]

\( \min C_x \) refers to the minimum difference between the range of each object and its mean value, and \( \min C_x \) is obtained corresponding to \( D_1 (D) \) is the initial cluster center \( O_1 \). Then, in order to eliminate the interference of outliers on the selection of cluster centers, a threshold value of \( \theta \) is set. When \( |V_l - \bar{V}| \leq \theta \), all the corresponding dataset objects \( D_1 \) in this range the candidate set \( A = \{A_1, A_2 \ldots A_l \ldots A_m\} \), finally, the maximum and minimum distance criterion in definition 3 is used to select the distance farthest from object \( O_1 \) is the second initial cluster center \( O_2 \), then:

\[
O_2 = \{A|\text{dis}(O_1, A_l)\} (l = 1, 2\ldots m) \tag{6}
\]

The initial cluster center \( O = \{O_1, O_2 \ldots O_K\} \) is obtained by analogy. In this way, the distance between the initial cluster centers is increased and the similarity is low. The calculation formula of range mean is as follows:

\[
\bar{V} = \frac{1}{n} \sum_{i=1}^{n} (\max d \text{ is}(x_i, y_j) - \min d \text{ is}(x_i, y_m)) \tag{7}
\]

Where \( \max d \text{ is}(x_i, y_j) \) and \( \min d \text{ is}(x_i, y_m) \) are the maximum and minimum value of Euclidean distance from other objects in set \( D \). \( \max d \text{ is}(x_i, y_j) - \min d \text{ is}(x_i, y_m) \) represents the extreme errors of object \( x_l \).

The initial clustering center will be in the relatively dense data objects, and the distance between them is far, which reduces the number of iterations of clustering and reduces the time complexity. At the same time, the interference of outliers is eliminated, so that the sum of squares of errors tends to be stable, and the time complexity of the algorithm is further reduced.

Proof 1: Set the point \( A, B, C \ldots \) in the set to be selected and outliers \( D, E, F \ldots A, B, C \ldots \) are far from \( D \), the number of \( D, E, F \ldots \) is far less than the concentration point.

\[
\therefore V_A = \max d \text{ is}(A, D) - \min d \text{ is}(A, B)
\]

And \( \because \text{dis}(A, B) \to 0 \)

\[
\therefore V_B = \max d \text{ is}(A, D) - \min d \text{ is}(A, B) = V_A
\]

In the same way \( V_C \approx V_A \).

By analogy, the range of the remaining \( n - 3 \) points also tend to this value.

And \( \because n >> 2 \) (2 only represents the minority)

\[
\therefore \bar{V} = \frac{1}{n} \sum_{n=1}^{n} V_n
\]

\( \therefore V_A \to \bar{V} \), the initial center point \( O \) is in the data object concentration.

If it is too far away from the group point, then first:

\[
\therefore V_D = \max d \text{ is}(D, A) - \min d \text{ is}(D, E), \text{ and is}(D, E) \gg \text{dis}(A, B)
\]

\( \therefore V_D < V_A \), and the range is large.

If it is too close to some points between the group points, \( \text{dis}(D, F) > \text{dis}(D, A), \text{dis}(D, E) \to \text{dis}(A, B) \), then 2:

\( V_D > V_A \) range is relatively large.

In summary, the range difference of \( |V_{D,E,F} - \bar{V}| >> |V_{A,B,C} - \bar{V}| \) outlier is larger than the absolute value of range mean.

Because the initial clustering centers are relatively dense and there is a certain distance between them, the sum of squares of clustering errors tends to be stable, the number of clustering iterations is greatly reduced, and the time complexity of the algorithm is reduced.
Because the range of outliers is larger than that of clustering points, so:

Definition 6: threshold $\theta$ comes from the range of the data object itself, that is, the purpose of threshold $\theta$ is to list the objects with the largest difference from the range of most objects as outliers, and exclude these points from the candidate set. The calculation formula is as follows:

$$
\theta = \frac{\sum_{i=1}^{n}((\max \text{dis}(x_i,y_i) - \min \text{dis}(x_i,y_m)) - \overline{V})}{n}
$$

(8)

The pseudo code of the operation process of range mean method is shown in algorithm 1.

---

**Algorithm 1 range mean method**

**Input:** data set $D$ ($n$ data objects, $m$ dimensions), clustering number $k$

**Output:** $k$ initial cluster centers

1. For each $D_i$ in I do
2. For each $D_{i+1}$ in I do
3. Calculate Euclidean distance of each data $d_{il}$; // Calculate the Euclidean distance between each data and other data,
4. Get$\{(\max (d_{il}), \min (d_{il}))\}$ // Get the maximum and minimum Euclidean distance between each data and other data
5. End for
6. $V = \max (d_{il}) - \min (d_{il})$
7. Get$\{V_0, V_1, V_2 ... V_{n-1}\}$ // Calculate the range and put it into the set
8. End for
9. Calculate mean of each $\{V_0, V_1, V_2 ... V_{n-1}\}$ $\overline{V}$; // calculate the average value $\overline{V}$ of set $\{V_0, V_1, V_2 ... V_{n-1}\}$
10. Get $\theta = \frac{\sum_{i=1}^{n}(V_i - \overline{V})}{n}$
11. For each $V_i$ in I do
12. $C_x = |V_i - \overline{V}|$
13. Get $C = \{C_0, C_1, C_2 ... C_{n-1}\}$
14. End for
15. For $C_i$ in I do
16. Choose the smallest of each $C_i$
17. Get $O = \{O_i\}$
18. If$C_i < \theta$
19. Get A.append$\{D_i\}$ // Add the data of the corresponding index to the candidate set $A$
20. End for
21. While $O$.len < $k$ do
22. Choose the max distance$\{O_0, O_1 ... O_{k-2}\}$ of each $A$ join $O$; // Select the data that is farthest from the existing object in the level $A$ to be selected and add it to the initial center point set $O$
23. End while
24. Return $O$

---

**2.3. Data Partition**

Mapper output data of MapReduce is divided and handed over to reducer for processing. The traditional partition method is the hash partition of MapReduce, that is, the keys with the same hash value are divided into the same reducer node for processing. There is a problem in this method: the
number of a key value under data skew is extremely large, which makes the reducer side overload. In order to solve this problem, this paper proposes BSA (based on pond sampling and first adaptation) based on pond sampling and first adaptation algorithm. Firstly, the number of middle nodes is calculated according to the sampling strategy of reducing algorithm. The specific steps are as follows:

(1) Sample forecast. The output data of mapper is sampled with equal probability by pond sampling method [11]. The number of sampling results is $I$, according to the sampling rate $p$, the total number of intermediate data is $M = \frac{I}{p}$, and the known number of nodes at the reducer end is $N$. Then the ideal capacity (the total number of objects with the same $K$) of each node at the reducer end is $S = \frac{M}{N}$, and the intermediate data output by mapper is $(key, value)$. It can be expressed as:

$$V = (k_1, v_1) (k_2, v_2) \ldots (k_i, v_i) \ldots (k_m, v_1) (k_m, v_2) \ldots (k_m, v_n)$$

(9)

Where $m$ is the number of different $K$, and $n$ is the number of values under $K$.

The pseudo code of sampling prediction process is shown in algorithm 2.

**Algorithm 2 Sampling prediction**

Input: sampling quantity $I$, intermediate data $V$, sampling rate $p$

Output: sample data $Y$, total number of intermediate data $M$

```
(1) Y = {};
(2) M = 0;
(3) For each $V_i$ do
(4) If($i < I$)
(5) Y.append($V_i$);
(6) Else
(7) $r$ = random.randint(0, $i$);
(8) If($r < I$)
(9) $Y[r] = V_i$;
(10) End for
(11) $M = Y, len / p$
(12) For each $Y_i$ in $i$ do
(13) Count the $n_i$ of number of the same $K_i$
(14) Get($n_1, n_2, n_3 \ldots n_i$)
Return $Y, M, (n_1, n_2, n_3 \ldots n_i)$
```

(2) Data scheduling. After getting the sampling result from algorithm 2, the sampling result $(n_1, n_2, n_3 \ldots n_i)$ according to the number of $K$ values. $k_i$ can be estimated according to the sampling rate. The total amount of intermediate data corresponding to $I$ is $M_{k_i} = \frac{I_{k_i}}{p}$. According to the ideal capacity $S$ of the node obtained above, the first adaptive scheduling is used to put the intermediate data into the node reasonably. From $k_i$ corresponds to the object with the most intermediate data, and starts to put the data into the remaining node that can hold it first. If the capacity of the remaining node is not enough to hold the $k_i$, the node with the largest capacity among the remaining nodes is selected to put in the data, and so on, until all the data is put in.

The pseudo code of data scheduling is as follows:

**Algorithm 3 Data scheduling**

```
(1) $Y = \{}$
(2) $M = 0$
(3) For each $V_i$ do
(4) If($i < I$)
(5) $Y.append(V_i)$;
(6) Else
(7) $r$ = random.randint(0, $i$);
(8) If($r < I$)
(9) $Y[r] = V_i$;
(10) End for
(11) $M = Y, len / p$
(12) For each $Y_i$ in $i$ do
(13) Count the $n_i$ of number of the same $K_i$
(14) Get($n_1, n_2, n_3 \ldots n_i$)
Return $Y, M, (n_1, n_2, n_3 \ldots n_i)$
```
Input: automatically sort the data cluster $M\{(k_i, m_i)\}$, sequence of Reducer nodes R, Ideal node capacity $S$
Output: partition scheme table $T(k_i, y)$

(1) RR=[] Remaining capacity of per node
(2) MR=[] The remaining objects of the class cluster corresponding to each $k$
(3) $T=${} Partition scheme table
(4) For $k \leftarrow 1$ to $N$ do
(5) $R_k = S$
(6) End for
(7) For $M_i, N_x$ in l,x do
(8) If(RR ≥ $M_i$)
(9) $T(k_i = M[l][1], y = x)$
(10) $RR_x = RR_x - M_i$// Get the remaining capacity of the x-th bucket
(11) Else
(12) $T(k_i = M[l][1], y = x)$
(13) $M$.append($k_i, M_1 - RR_x$))
(14) End for
Return $T(k_i, y)$;

2.4. Parallelization
After getting the initial clustering center, in order to mine the data center in parallel, the final clustering result is obtained. It needs to go through two stages: Map and Reduce. In the Map stage, the initial center point set in algorithm 1 or the result of the last job is called. The distance between each data object and the center point of the cluster is calculated according to the Euclidean distance formula in definition 1, and then the objects are divided into corresponding clusters according to the distance size, and the results are saved in Combine. Before the Reducer stage, the data in the Map stage should be divided and scheduled. According to the scheduling table obtained by the sampling prediction and the first adaptation algorithm, the data is divided into each node. In the Reduce stage, local clusters are merged, and a new cluster center is calculated to determine whether the global optimal solution is obtained. If not, a new round of MapReduce task is continued until the global optimal solution is obtained and the clustering is finished.

Algorithm 4 Parallel clustering
Input: initial cluster center, partition scheme table
Output: final clustering results

(1) Enable MapReduce
(2) Map(key, value)
(3) Input $O$ and $\{R_0, R_1, \cdots, R_{r-1}\}$;
(4) Get the local clustering results; // The local clustering results are obtained according to Euclidean distance;
(5) Deposit in Combine; // The results are temporarily stored in Combine
(6) End
(7) Reduce (key, value)
(8) Input $T(k_i, y)$
(9) Read Combine; // Access intermediate data according to partition scheme table
(10) If $E =_{\min}$ false do
(11) Enable a new MapReduce task; // Enable new MapReduce task
(12) Else do
(13) Get the final clustering results; // Obtain the final clustering results
(14) End
(15) End
Return the final clustering results;

2.5. KMEANS-BRMS Algorithm Steps
The implementation steps of KMEANS-BRMS algorithm are as follows:

Step 1: input the data set \( D \) to be clustered and call the range mean method of algorithm 1 to obtain the initial cluster center.

Step 2: in the intermediate data division stage, first call algorithm 2 "sampling prediction" to get the total number of sampling data and intermediate data, and then call algorithm 3 "data scheduling" to get the partition scheme table.

Step 3: reads the initial cluster center and partition scheme table, starts MapReduce task, calls algorithm 4 for parallel partition clustering, and obtains the final clustering result.

2.6. Algorithm Analysis
KMEANS-BRMS algorithm is mainly composed of the selection of initial cluster center, data partition and parallel clustering, and its time complexity is mainly affected by these three parts: (1) The initial cluster center selection phase. Firstly, the Euclidean distance from each point to other points in the space is calculated, and the maximum and minimum values are obtained, and the time complexity is \( O(n^2) \), secondly, the mean value of range is calculated, and the time complexity is \( O(n) \), so the time complexity of this stage is \( O(n^2 + n) \); (2) In the stage of data partition, the first step is to sample the intermediate data. The time complexity of the method is \( O(n) \), and the second step is sorting scheduling. Assuming that the number of nodes is \( n \), the time complexity of this stage is \( O(2N) \) because it needs to traverse all nodes and record the remaining capacity, and then traverse the set of the remaining capacity \( O(n + 2N) \); (3) Parallel clustering stage. In this stage, before the final clustering result is obtained, MapReduce task needs to be called continuously to update the cluster center, and the time complexity is \( O(n \log n) \). In conclusion, the time complexity of KMEANS-BRMS algorithm is \( O(n^2 + n \log n + 2n + 2N) \). Because \( N \ll n \), the time complexity of KMEANS-BRMS is approximately \( O(n^2) \).

3. Analysis of Experimental Results
3.1. Experimental Environment
In order to verify the performance of the algorithm, this experiment calls several real datasets from UCI public database to experiment in Hadoop cluster. The experiment is a Hadoop cluster composed of one master node and three slave nodes [16]. The nodes are connected through 200MB/s Internet. The configuration of all nodes is the same as follows: hard disk 512Gb, memory 8GB, CPU inter core i5-8750H, operating system Ubuntu 15.02 [17, 18], Hadoop version installed in the system is 2.6.1, and the experiment is compiled with Java JDK 1.8.4.

3.2. Data Sets
The experimental data of KMEANS-BRMS algorithm is from UCI public database, Iris, Poker Hand, PPG-DaliA and Sift10M from simple to complex. Iris data set has 150 pieces of data, is a classic small data set; Poker Hand data set has 1025010 pieces of data, comes from playing cards; PPG DaliA data set has 8.3 million pieces of data, is a PPG based mind estimation data set; Sift10M data set has 11164866 pieces of data, is a large data set, with many attribute categories. The details of the dataset are shown in Table 1.
Table 1. Experimental data set information

| Data Set       | Records number | Attribute | Label | Size(MB) |
|---------------|---------------|-----------|-------|----------|
| Iris          | 150           | 4         | 3     | 0.86     |
| Poker Hand    | 1025010       | 11        | 3     | 235      |
| PPG-DaliA     | 8300000       | 11        | 15    | 2700     |
| Sift10M       | 11164866      | 128       | 2     | 7100     |

3.3. Algorithm Performance Analysis
In order to test the performance of KMEANS-BRMS algorithm in large data sets, the speedup ratio is used to measure the parallel computing time of the algorithm. The algorithm uses several experiments to evaluate the comprehensive performance of kmeans-brms algorithm in Iris, Poker Hand, PPG-DaliA and Sift10M data sets. The average speedup curve of the KMEANS-BRMS algorithm is shown in the figure.

Figure 1. Speedup curve of KMEANS-BRMS algorithm

It can be seen from Figure 1 that KMEANS-BRMS algorithm has better speedup performance in large datasets, while in small datasets, such as iris datasets, the speedup performance decreases and is less than 1 due to the increase of the number of nodes. This is because Iris datasets are too small. The time cost between nodes caused by the increase of nodes is greater than the acceleration provided by the increase of nodes time. In a small data set, too many nodes will make many nodes idle, but the communication between them is very busy. When dealing with large data sets, because of the large data sets, adding nodes can effectively speed up the calculation time, and the communication cost between nodes can be ignored. As shown in Figure 1, when processing the largest data set of Sift10M, the speedup of each additional node is significantly higher than that of the smaller data set. When processing four nodes, the speedup exceeds 4, which indicates that KMEANS-BRMS algorithm has more obvious advantages in processing the larger data set. With the increase of the number of nodes, the speedup ratio increases approximately linearly. To sum up, KMEANS-BRMS algorithm has a significant effect in large data set clustering, and the effect will be more obvious with the increase of node number.

3.4. Algorithm Performance Comparison
In addition to verifying the effect of parallelization, it is also necessary to evaluate the clustering students. The commonly used indicators are the optimal value of clustering results, accuracy, variance, running time, etc. In this paper, comparative experiments are carried out on three large datasets: Poker Hand, PPG-DaliA and Sift10M. The performance of each algorithm is compared with the algorithms in literature 4 and 6 respectively. Each algorithm runs ten times and takes the average value. As shown in Table 2, the variance, optimal value and accuracy of each algorithm are compared.
Table 2. Algorithmic clustering comparative analysis

| Data set  | Algorithm       | Optimal value | Variance  | Accuracy /% |
|-----------|-----------------|---------------|-----------|-------------|
| Poker Hand| Reference 4     | 3.575E+9      | 3.612E-6  | 74.2        |
|           | Reference 6     | 2.212E+7      | 1.245E-8  | 92.6        |
|           | KMEANS-BRMS     | 2.025E+7      | 1.195E-9  | 93.7        |
| PPG-DaliA | Reference 4     | 4.585E+7      | 3.778E-6  | 77.9        |
|           | Reference 6     | 3.926E+6      | 1.798E-8  | 93.1        |
|           | KMEANS-BRMS     | 3.660E+6      | 3.717E-9  | 94.2        |
| Sift10M   | Reference 4     | 5.508E+6      | 3.954E-6  | 76.3        |
|           | Reference 6     | 2.679E+6      | 2.026E-8  | 91.2        |
|           | KMEANS-BRMS     | 2.157E+6      | 1.907E-9  | 93.3        |

The running time can reflect the time complexity, as shown in Figure 2.

Figure 2. The execution time of each algorithm under the three data sets
It can be analyzed from table 2 that the accuracy rate of literature 6 algorithm is better than that of literature 4 algorithm in each data set, and the accuracy rate of Poker Hand is improved by 18.4%. This is because literature 6 algorithm uses genetic algorithm to optimize the selection of initial center, avoiding the adverse impact of random selection of initial cluster center. The accuracy of this algorithm is still above the two, especially in the Sift10M data set, the optimal value is relatively low. This shows that KMEANS-BRMS algorithm has more advantages than the above two algorithms.

It can be seen from Figure 2 that the running time of KMEANS-BRMS algorithm under different number of nodes is lower than that of literature 4 algorithm and literature 6 algorithm. This is because KMEANS-BRMS uses the range method to select the initial cluster center, which eliminates the interference of outliers and increases the diversity of objects between clusters. At the same time, it uses the BSA strategy for the partition scheduling of intermediate data, which avoids the data skew problem and improves the efficiency. The utilization of the points reduces the running time of the algorithm.

4. Summary
In order to solve the shortcomings of traditional K-means algorithm under the background of big data, this paper proposes an optimization algorithm of K-means algorithm based on range mean and sampling: KMEANS-BRMS. In order to verify the superiority of KMEANS-BRMS algorithm, the algorithm in this paper and the three algorithms in literature 4 and 6 are tested in Iris, Poker Hand, PPG DaliA and Sift10M. The experimental results show that the clustering effect of KMEANS-BRMS algorithm in large-scale data sets is significantly improved, and the parallel efficiency is also significantly increased.

References
[1] Yang Junchuang, Zhao Chao. Survey on K-means clustering algorithm. Computer Engineering and Applications, 2019, 55(23):7-14+63.
[2] Mai X, Cheng J, Wang S. Research on semi supervised K-means clustering algorithm in data mining. Cluster Computing, 2018.
[3] Ren Yuanhang, Literature. Survey of K-means algorithms on big data [J/OL]. Application Research of Computers: 1-7[2020-11-05].
[4] Zang Yanhui, Xi Yunjiang, Zhao Xuezhang. Divide and conquer k-means clustering method based on MapReduce. Computer Engineering and Design, 2020, 41(05):1345-1351.
[5] Cai S, Sun R, Hao S, et al. An Efficient Outlier Detection Approach on Weighted Data Stream Based on Minimal Rare Pattern Mining. China Communications, 2019(10).
[6] Lai Xiangyang, Gong Xiujun, Han Laiming. Genetic algorithm based K-medoids clustering within MapReduce framework. Computer Science, 2017, 44(03):23-26+58.
[7] Ali F, Atila U, Abu-Issa R, et al. A novel data clustering algorithm based on gravity center methodology. Expert Systems with Applications, 2020.
[8] Yang P, Wang D, Wei Z, et al. An Outlier Detection Approach Based on Improved Self-Organizing Feature Map Clustering Algorithm. IEEE Access, 2019.
[9] Wantao Ning. Connectivity and super connectivity of the divide-and-swap cube. 2020, 842:1-5.
[10] Liu Min, Li Xu. UEL program realization of two-dimensional secondary interface element based on cohesion theory in ABAQUS. Chinese Journal of Computational Mechanics, 2019, 36(05):693-698.
[11] Tao Yongca, Ding Leidao, Shi Lei, Wei Lin. Research on MapReduce On-line Load Balancing Based on Sample Partition. Journal of Chinese Computer Systems, 2020, 46(08):46-53.
[12] Benelallam A, Gómez, Abel, Tisi M, et al. Distributing Relational Model Transformationon MapReduce. Journal of Systems and Software, 2018: S0164121218300669.
[13] Sun Yuqiang, Li Yuanyuan, Lu Yong. Optimization of K-means clustering algorithm based on MapReduce. Computer Measurement & Control, 2016,24(07):272-275+279.
[14] Huang Honghua. BF Algorithm for Packing Problem Based on Visual C++. Computer
Knowledge and Technology, 2018, 14(36):258-259.

[15] Kłopotek Mieczysław A. On the Consistency of k-means++ algorithm. 2020, 172(4):361-377.

[16] Dongbo Zhang, Yanfang Shou, Jianmin Xu. An improved parallel K-means algorithm based on MapReduce. 2017, 9(3):275-282.

[17] Tom Henderson, Tom Henderson. Ubuntu 16.04 kisses the cloud, disses the desktop. 2016,

[18] Yingjun Tan, Cuixia Li, Madison Murphy. Optimization of Multimedia Data Clustering Method under Spark Cloud Computing Platform. 2016, 9(2):83-87.