A Parallel Clustering Method Study Based on MapReduce

Sun Zhanquan[1], Geoffrey Fox[2]

(1 Key Laboratory for Computer Network of Shandong Province, Shandong Computer Science Center, Jinan, Shandong, 250014, China

2 School of Informatics and Computing, Pervasive Technology Institute, Indiana University Bloomington, Bloomington, Indiana, 47408, USA)

Sun30@indiana.edu, gcf@indiana.edu

Abstract: Clustering is considered as the most important task in data mining. The goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. Many practical application problems should be solved with clustering method. It has been widely applied into all kinds of areas, such marketing, biology, library, insurance, earth-quake study, and World Wide Web and so on. Many clustering methods have been studied, such as k-means, Fisher clustering, and Koehon clustering and so on. In many kinds of areas, the scale of data set becomes larger and larger. Classical clustering method will not work to deal with large scale data set. The study of clustering methods based on large scale data is considered as an important task. MapReduce is taken as the most efficient model to deal with data intensive problems. Many data mining methods based on MapReduce have been studied. In this paper, parallel clustering method based on MapReduce is studied. The research mainly contributes the following aspects. Firstly, it determines the initial center objectively. Secondly, information loss is taken as the distance metric between two samples. Thirdly, the clustering results are visualized with interpolation MDS method. The efficiency of the method is illustrated with a practical DNA clustering problem.

Keywords: Information bottleneck theory, MapReduce; Twister

1 Introduction

With the development of electronic and computer technology, the quantity of electronic data is in exponential growth [1]. Data deluge has become a salient problem to be solved. Scientists are overwhelmed with the increasing amount of data processing needs arising from the storm of data that is flowing through virtually every science field, such as bioinformatics [2-3], biomedical [4-5], Cheminformatics [6], web [7] and so on. Then how to take full use of these large scale data to support decision is a big problem encountered by scientists. Data mining is the process of discovering new patterns from large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics and database systems. It has been studied by many scholars in all kinds of application area for many years and many data mining methods have been developed and applied to practice. But most classical data mining methods out of reach in practice in face of big data. Computation and data intensive scientific data analyses are increasingly prevalent in recent years. Efficient parallel/concurrent algorithms and implementation techniques are the key to meeting the scalability and performance requirements entailed in such large scale data mining analyses. Many parallel algorithms are implemented using different parallelization techniques such as threads, MPI, MapReduce, and mash-up or workflow technologies yielding different performance and usability characteristics [8]. MPI model is efficient in computation intensive problems, especially in simulation. But it is not easy to be used in practical. MapReduce is a cloud technology developed from the data analysis model of the information retrieval field. Several MapReduce architectures are developed now. The most famous is the Google, but the source code is not open. Hadoop is the most popular open source MapReduce software. It has been adopted by many huge IT companies, such as Yahoo, Facebook, eBay and so on. The MapReduce architecture in Hadoop doesn't support iterative Map and Reduce tasks, which is required in many data mining algorithms. Professor Fox developed an iterative MapReduce architecture software Twister. It supports not only non-iterative MapReduce applications but also an iterative MapReduce programming model. The manner of Twister MapReduce is "configure once, and run many time" [9-10]. It can be applied on cloud platform. It will be the popular MapReduce architecture in cloud computing and can be used in data intensive data mining problems.

Clustering is a main task of explorative data mining, and a common technique for statistical data analysis used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. The goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. Many classical clustering methods have been studied and widely applied to many kinds of field, such as k-means, Fisher clustering method, Kohonen neural network and so on[11-12]. Information bottleneck (IB) theory is proposed by

Tishby [13]. It is a data compression method based on Shannon's rate distortion theory. The clustering method based on IB theory was widely studied in recent years. It measures the distance between samples with the quantity of information loss caused by merging. It has been applied to the clustering of image, texture, and galaxy successfully [14-15] and got good results. But when the scale of data set becomes larger and larger, classical clustering method will not work to deal with large scale data set. How to develop clustering methods based MapReduce to process large scale data is an important issue. It is the development intention of big data science. Many scholars have done lots work on this topic. Some clustering methods based on MapReduce were proposed, such as k-means, EM, Dirichlet Process Clustering and so on. Though the clustering method based on IB theory is efficient in processing complicated clustering problem, it can't be transformed to MapReduce model directly. Centroid-based clustering is a kind of important clustering method. There exit two problems in the centroid-based clustering method to be resolved. The first one is that the initial centroid will affect the final clustering results. There is not an objective method to determine the initial center. Another one is that the distance measure will have great effect on the clustering result. Most distance measure can't describe the complicated correlation between samples.

The evaluation of unsupervised clustering result is a difficult problem. Visualization is a good mean to For improving the computation speed of SVM, feature extraction is an efficient means. In practical, there are many problems' feature variable vector is in high dimension. Too many input variable will increase the computation cost of SVM. Feature extraction can decrease the dimension of input and decrease the computation cost efficiently. Many feature extraction methods have been proposed, such as Principal Component Analysis (PCA), Self Organization Map (SOM) network, and so on[16-17]. Multidimentional Scaling (MDS) is a kind of Graphical representations method of multivariate data[18]. It is widely used in research and applications of many disciplines. The method is based on techniques of representing a set of observations by a set of points in a low-dimensional real Euclidean vector space, so that observations that are similar to one another are represented by points that are close together. It is a nonlinear dimension reduction method. But the computation complexity is $O(n^2)$ and memory requirement is $O(n^2)$. With the increase of sample size, the computation cost of MDS increase sharply. For improving the computation speed, interpolation MDS are introduced in reference [19]. It is used to extract feature from large scale data. In this paper, interpolation MDS is combined with parallel SVM based on MapReduce to analyze large scale data.

In this paper, a novel clustering method based on MapReduce is proposed. It combines IB theory with centroid based clustering method. Firstly, IB theory based hierarchy clustering is used determine the center of each Map computation node. All sub-controids are combined into one centroid with the IB theory also in Reduce computation node. For measure the complicated correlation between samples, information loss is used to measure the distance. The clustering method is an iterative model. The clustering method is programmed with iterative MapReduce model Twister. For showing the clustering results, interpolation MDS is used to reduce the samples into 3 dimensions. The reduced clustering results are show in 3D coordination with Pviz software developed by Indiana University. Bioinformatics is an important large scale data analysis. Lots of bioinformatics data will be generated all over the work. DNA clustering is an important task. This paper will take DNA clustering as an example to illustrate the efficiency of the proposed clustering method.

The following of the paper is organized as follows. IB theory will be introduced in section 2. The clustering method based on MapReduce will be described in detail in section 3. Interpolation MDS dimension reduction method is introduced in section 4. DNA clustering problem is presented in section 5. A DNA analysis example is analyzed in section 6. At last, some conclusions are drawn.

2 IB Principle

The IB clustering method states that among all the possible clusterings of a given object set when the number of clusters is fixed, the desired clustering is the one that minimizes the loss of mutual information between the objects and the features extracted from them. Assume there is a joint distribution $p(x, y)$ on the "object" space X and the "feature" space Y. According to the IB principle we seek a clustering $\hat{X}$ such that the information loss $I(X;\hat{X}) = I(X;Y) - I(\hat{X};Y)$ is minimized. $I(X;\hat{X})$ is the mutual information between X and $\hat{X}$

$$I(X;\hat{X}) = \sum_{x,\hat{x}} p(x)p(\hat{x}\mid x)\log\frac{p(\hat{x}\mid x)}{p(\hat{x})} \tag{1}$$

The IB principle is motivated from Shannon's rate-distortion theory which provides lower bounds on the number of classes. Given a random variable X and a distortion $d(x_1, x_2)$ measure, we want to represent the symbols of X with no more than R bits. The rate-distortion function is given

$$D(R) = \min_{p(\hat{x}|x)|I(X;\hat{X})\le R} Ed(x,\hat{x}) \tag{2}$$

where $Ed(x,\hat{x}) = \sum_{x,\hat{x}} p(x)p(\hat{x}\mid x)d(x,\hat{x})$.

The loss of the mutual information between X and Y caused by the clustering $\hat{X}$ can be viewed as the average of this distortion measure

$$\begin{aligned} d(x,\hat{x}) &= I(X;Y) - I(\hat{X};Y) \\ &= \sum_{x,\hat{x},y} p(x,\hat{x},y)\log\frac{p(y\mid x)}{p(x)} - \sum_{x,\hat{x},y} p(x,\hat{x},y)\log\frac{p(y\mid \hat{x})}{p(y)} \\ &= ED(p(x,\hat{x}) \parallel p(y\mid \hat{x})) \end{aligned} \tag{3}$$

where $D(f \parallel g) = E_f \log(f/g)$ is the KL divergence. We can obtain the rate distortion function

$$D(R) = \min_{p(\hat{x}|x)|I(X;\hat{X})\le R} (I(X;Y) - I(\hat{X};Y)) \tag{4}$$

which is exactly the minimization criterion proposed by the IB principle, i.e. finding a clustering that minimize the loss of mutual information between the objects and the features.

Let c_1 and c_2 be two clusters of symbols, the information loss due to the merging is

$$d(c_1,c_2) = I(c_1;Y) + I(c_2;Y) - I(c_1,c_2;Y) \tag{5}$$

Standard information theory operation reveals

$$d(c_1,c_2) = \sum_{y,i=1,2} p(c_i)p(y\mid c_i)\log\frac{p(y\mid c_i)}{p(y\mid c_1\cup c_2)} \tag{6}$$

where $p(c_i) = \mid c_i \mid / \mid X \mid$, $\mid c_i \mid$ denotes the cardinality of c_i , $\mid X \mid$ denotes the cardinality of object space X , $p(c_1\cup c_2) = \mid c_1\cup c_2 \mid / \mid X \mid$.

It assumes that the two clusters are independent when the probability distribution is combined. The combined probability of the two clusters is

$$p(y\mid c_1\cup c_2) = \sum_{i=1,2}\frac{|c_i|}{|c_1\cup c_2|}p(y\mid c_i) \tag{7}$$

The minimization problem can be approximated by a greedy algorithm based on a bottom-up merging procedure. The algorithm starts with the trivial clustering where each cluster consists of a single data vector. In order to minimize the overall information loss caused by the clustering, classes are merged in every step, such that the information loss caused by merging them is the smallest. The method is suitable to both sample clustering and feature clustering. Due to the sample are sequence, it is not permit to break the sequence during the clustering procedure. The procedure serial clustering based on IB can be summarized as follows.

(1) Start with the trivial clustering where each cluster consists of a single data vector.
(2) Calculate the information loss caused by the merging of each two neighboring classes according to 6). Choose the pair of classes with the minimum information loss and merge them into one class. After merging, the two classes are taken as one class and the ratios are calculated according to (7).
(3) Iterate step 2 until all the data vectors merge into one class.
(4) The number of clusters can be determined according to the information loss of each step. The information loss will increase with the meager step. The rule is that the merger will stop when the cluster number equal to the prescribed cluster number.

3 Clustering Based on MapReduce

3.1 Architecture of Twister

There are many parallel algorithms with simple iterative structures. Most of them can be found in the domains such as data clustering, dimension reduction, link analysis, machine learning, and computer vision. These algorithms can be implemented with iterative MapReduce computation. Professor Fox developed the first iterative MapReduce computation model Twister. It has several components, i.e. MapReduce main job, Map job, Reduce job, and combine job. Twister's programming model can be described as in figure 1.

MapReduce jobs are controlled by the client node through a multi-step process. During configuration, the client assigns MapReduce methods to the job, prepares KeyValue pairs and prepares static data for MapReduce tasks through the partition file if required. Between iterations, the client receives results collected by the Combine method,

and, when the job is done, exits gracefully. The message communicate between job is realized with message brokers, i.e. NaradaBrokering or ActiveMQ.

Map daemons operate on computation nodes, loading the Map classes and starting them as Map workers. During initialization, Map workers load static data from the local disk according to records in the partition file and cache the data into memory. Most computation tasks defined by the users are executed in the Map workers. Twister uses static scheduling for workers in order to take advantage of the local data cache. In this hybrid computing model, daemons communicate with the client through messages.

Reduce daemons operate on computation nodes. The number of reducers is prescribed in client configuration step. The reduce jobs depend on the computation results of Map jobs. The communication between daemons is through messages.

Combine job is to collect MapReduce results. It operates on client node. Twister uses scripts to operate on static input data and some output data on local disks in order to simulate some characteristics of distributed file systems. In these scripts, Twister parallel distributes static data to compute nodes and create partition file by invoking Java classes. For data which are output to the local disks, Twister uses scripts to gather data from all compute nodes on a single node specified by the user.

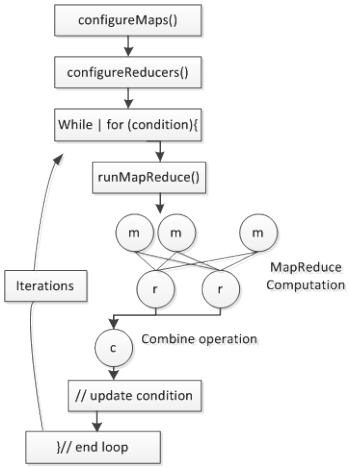

Figure 1 Twister's programming model

3.2 Clustering based on Twister

The parallel clustering method can be divided into two parts. The first part is to determine the initial center point. The second part is to obtain the global center point through iteration and get the final clustering results.

3.2.1 Initial centroid calculation

Given data set D with n samples, it is divided into m partitions $D^1, D^2, \cdots, D^m$ with $n_1, n_2, \cdots, n_m$ samples separately. Operate clustering on each partition $D^i = \{D_1^i, D_2^i, \cdots, D_{n_i}^i\}, i = 1, \cdots, m$ with the clustering method introduced in section 2. We can obtain the sub-centroids $C^i = \{C_1^i, C_2^i, \cdots, C_{n_i}^i\}, i = 1, \cdots, m$. All sub-centroids are collected together to generate new data set $C = \{C^1, C^2, \cdots, C^m\}$. Apply clustering based on information bottleneck theory to the new dataset. Then we can obtain the initial global center C^0. In the calculation equation (7), the number of each clustering is considered. The centroid vector should include the numbers of samples that generate the centroid. The realization of the calculation process based on Twister is shown in figure 2. Firstly, partitioned datasets are distributed to each computation node. In each Map computation node, operate IB on each dataset to obtain the sub-centroid. All sub-centroids are collected in Reduce node to generate new dataset. Apply IB on the new dataset to generate the initial centroid of the global dataset.

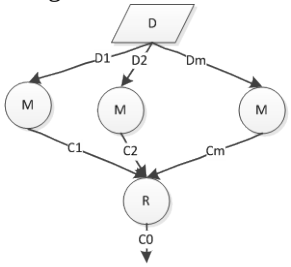

Fig. 2 initial centroid calculation process

3.2.2 Centroid based iterative clustering

After getting the initial center C^0, it is used to calculate the final centroid. The process is as follows. For each sample in each sub dataset $\boldsymbol{x} \in D^i$, calculate the distance between the sample and all the samples in center data set C^0. In the calculation, information loss (6) is taken as the distance measure. Set k empty dataset $P^1, P^2, \cdots, P^k$. The sample will be added to dataset P^i if the distance between x and the center c_i^0 is minimum. Recalculate the center of C^i with the dataset P^i according to Eq. (7). After calculating the new sub-centroids $C^1, C^2, \cdots, C^m$, calculate the new centroid C^0 according to the following equation.

$$c_i^0 = \sum_{j=1}^{k} \frac{|C^j|}{|C^1 \cup C^2 \cdots C^k|} c_i^j \tag{8}$$

Through calculation the difference between the old C^0 and the new generated C^0 to determine whether the iteration will stop. The iteration process based on Twister is shown as in figure 3. The samples have be partitioned and deployed in each computation node in the first step. The initial C^0 get from the first step is mapped to each computation node. In each Map node, recalculate the sub-centroids. All sub-centroids are collected in Reduce node and regenerate the global centroid C^0 according to (8). The new centroid are feedback to main computation node and calculate the difference between the old C^0 and the new generated C^0. Iteration will stop when the difference is less than the prescribed threshold value.

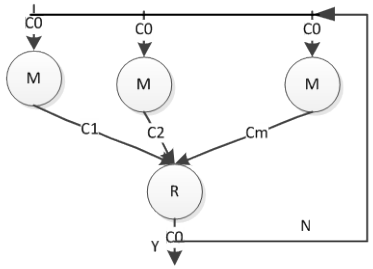

Fig. 3 iteration process to calculate final centroid

4 Interpolation MDS

To visualize the clustering results, the high dimension samples should be mapped into 3 dimensions. MDS is an efficient dimension reduction method. It is as follows.

4.1 Multidimensional Scaling

MDS is a non-linear optimization approach constructing a lower dimensional mapping of high dimensional data with respect to the given proximity information based on objective functions. It is an efficient feature extraction method. The method can be described as follows.

Given a collection of n objects $\mathrm{D} = \{\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_\mathrm{n}\}, \mathrm{x}_\mathrm{i} \in \mathrm{R}^\mathrm{N}(\mathrm{i} = 1,2, \cdots, \mathrm{n})$ on which a distance function is defined as $\delta_{\mathrm{i,j}}$, the pairwise distance matrix of the n objects can be denoted by

$$\Delta := \begin{pmatrix} \delta_{1,1} & \delta_{1,2} & \cdots & \delta_{1,\mathrm{n}} \\ \delta_{2,1} & \delta_{2,2} & & \delta_{2,\mathrm{n}} \\ \vdots & & \ddots & \vdots \\ \delta_{\mathrm{n},1} & \delta_{\mathrm{n},2} & \cdots & \delta_{\mathrm{n,n}} \end{pmatrix}$$

where $\delta_{\mathrm{i,j}}$ is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Euclidean distance is often adopted.

The goal of MDS is, given Δ, to find n vectors $\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n \in R^L (L \leq N)$ to minimization the STRESS or SSTRESS. The definition of STRESS and SSTRESS are as follows.

$$\sigma(P) = \sum_{i<j} w_{i,j} \left(d_{i,j}(P) - \delta_{i,j}\right)^2 \tag{9}$$

$$\sigma^2(P) = \sum_{i<j} w_{i,j} \left((d_{i,j}(P))^2 - \delta_{i,j}^2\right)^2 \tag{10}$$

where $1 \leq \mathrm{i} < \mathrm{j} \leq \mathrm{n}$, $w_{i,j}$ is a weight value ($w_{i,j} > 0$), $d_{i,j}(P)$ is a Euclidean distance between mapping results of $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. It may be a metric or arbitrary distance function. In other words, MDS attempts to find an embedding from the n objects into R^L such that distances are preserved.

4.2 Interpolation Multidimensional Scaling

One of the main limitations of most MDS applications is that it requires $O(n^2)$ memory as well as $\mathrm{O}(\mathrm{n}^2)$ computation. It is difficult to process MDS with large scale data set because of the limitation of memory limitation. Interpolation is a suitable solution for large scale MDS problems. The process can be summarized as follows.

Given n samples data $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}, \boldsymbol{x}_i \in R^N (i = 1,2, \cdots, n)$ in N dimension space, m samples $D_{sel} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\}$, are selected to be mapped into L dimension space $P_{sel} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_m\}$ with MDS. The other samples $D_{rest} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{n-m}\}$, will be mapped into L dimension space $P_{rest} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_{n-m}\}$ with interpolation method. The computation cost and memory of interpolation MDS is only $O(n)$. It can improve the computing speed markedly.

Select one sample data $\boldsymbol{x} \in D_{rest}$, calculate the distance δ_{ix} between the sample data $\boldsymbol{x}$ and the pre-mapped samples $\boldsymbol{x_i} \in D_{sel} (i = 1,2, \cdots, m)$. Select the k nearest neighbors $Q = \{q_1, q_2, \cdots, q_k\}$, where $\boldsymbol{q}_i \in D_{sel}$, who have the minimum distance values.

After data set Q being selected, the mapped value of the input sample is calculated through minimizing the following equations as similar as normal MDS problem with $k + 1$ points.

$$\sigma(X) = \sum_{i<j} \left(d_{i,j}(P) - \boldsymbol{\delta_{i,j}}\right)^2 = C + \sum_{i=1}^{k} d_{ip}^2 - 2\sum_{i=1}^{k} d_{ip}\delta_{ix} \tag{11}$$

In the optimization problems, only the position of the mapping position of input sample is variable. According to reference [10], the solution to the optimization problem can be obtained as

$$x^{[t]} = \overline{\boldsymbol{p}} + \frac{1}{k}\sum_{i=1}^{k} \frac{\delta_{ix}}{d_{iz}} \left(x^{[t-1]} - \boldsymbol{p}_i\right) \tag{12}$$

where $d_{iz} = \left\|\boldsymbol{p}_i - x^{[t-1]}\right\|$ and $\overline{\boldsymbol{p}}$ is the average of k pre-mapped results. The equation can be solved through iteration. The iteration will stop when the difference between two iterations is less than the prescribed threshold values. The difference between two iterations is denoted by

$$\delta = \frac{(\|x^{[t]} - x^{[t-1]}\|)}{\|x^{[t-1]}\|} \tag{13}$$

5 DNA sequence clustering

Generally, A, C, T, G letters are used to denote a DNA sequence. A DNA sequence can be taken as a nonempty string S of letter set $\sum(\sum = \{A, C, T, G\})$, i.e. $S = (s_1, s_2, \cdots, s_n)$, where $n = |S| > 0$ denotes the length of the string. A DNA can be expressed with the frequency character of 4 letters $\{A, C, T, G\}$ and the frequency distribution of double sequence nucleic acid, i.e. adjacent two nucleic acids are composed into a string. The frequency character of double sequence nucleic acid extracted from a DNA sequence can compose a 16 dimension vector $[AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT]$. The frequency of each vector can be calculated as the formula

$$f_{s_i,s_j} = \frac{S_i S_j}{|S|-1} \tag{14}$$

where $s_i s_j \in \sum$, $S_i S_j$ denote the frequency of some double sequence nucleic acid in a DNA string. $|S|$ denotes the length of DNA sequence. In the above equation, the nucleic acids exclude the head and end of the string are calculated two times. For removing the effect of single nucleic acid, the frequency of double nucleic acid is modified by

$$p_{s_i,s_j} = \frac{f_{s_i,s_j}}{f_{s_i} f_{s_j}} \tag{15}$$

For calculating the information loss, the frequency should be normalized, i.e.

$$p^*_{s_i,s_j} = \frac{p_{s_i,s_j}}{\sum p_{s_i,s_j}} \tag{16}$$

The clustering based on information bottleneck theory can be summarized as follows.

1) Extract frequency feature from DNA string and generate frequency vectors of samples.
2) Divide DNA samples into N partitions. Configure the MapReduce environment and put the N partitions to m computation nodes.
3) Run the clustering based on MapReduce according to section 4.

6 Examples

6.1 Data source

The initial data set was received from Dr. Mina Rho in Indiana University. They are some 16S rRNA data. We select 100043 DNA data to clustering analysis. In the data file, each DNA record is expressed with a G, A, C, and T strings.

6.2 Preprocess

Calculate the probabilities of $\{A, C, T, G\}$ and $[AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA, TC, TG, TT]$ of each string. Then calculate the modified frequency according to Eq. (15). At last, the frequencies are normalized according to Eq.(16). The sample strings are transformed into 16 dimensions vector. They are described with probabilities are taken as the input of clustering.

6.3 DNA clustering

The initial data set are partitioned into 100 partitions. They are deployed to 8 computation nodes. Apply IB theory to each data set partition. Get 100 sub-centroids. One Reduce computation point is used to combine all the sub-controids into one centroid with IB theory. They are taken the initial centroid of the centroid clustering method. They are mapped to each computation node. Recalculate the centroid of the partition according to the section 4.2 iteratively. When the stop rule is met, the clustering stops. The computation times based on different cluster number are listed in table1.

Table 1 computation time based on different cluster number

Centroid number	Partition number	Computation nodes	Computation time
3	100	8	14465.368
5	100	8	14512.32
10	100	8	15132.52

6.4 Clustering result showing

In this example, 4000 samples are selected to be pre-mapped into 3 dimension space. Firstly, calculate the distance matrix. Euclidean distance is adopted here. Then calculate the mapped vector according to the distance matrix with MDS method. The others are mapped into low dimension with interpolation MDS method. The number of nearest neighbor is set $k = 10$. After dimension reduction, the clustering results are shown as in figure 3, 4, and 5 respectively.

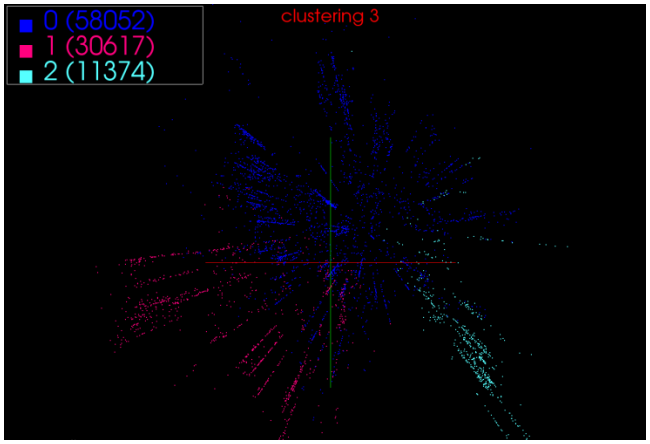

Fig. 3 clustering results of 3 clusters

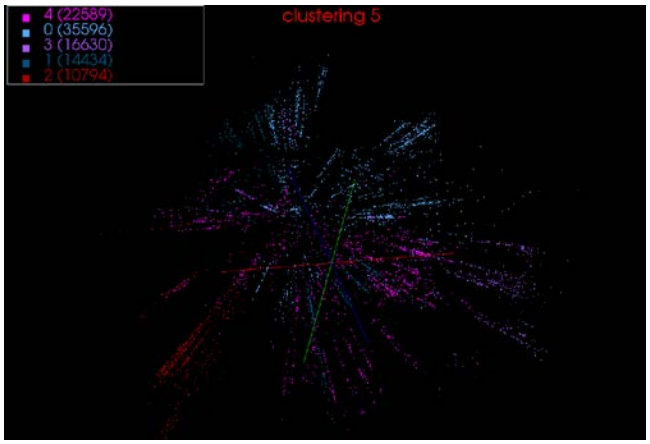

Fig.4 clustering results of 5 clusters

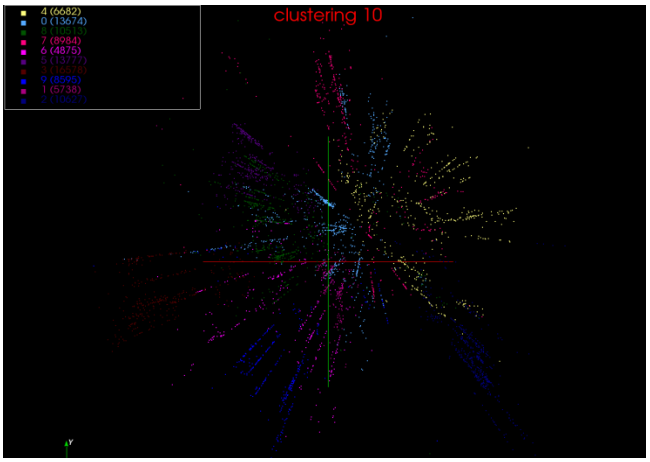

Fig.5 clustering results of 10 clusters

7 conclusions

Large scale data clustering is an important task in many application fields. The proposed clustering method based on MapReduce is an efficient method for large scale data analysis. It provides an objective method to determine the initial clustering centroid. The information loss is used to measure the distance between samples. It can measure any complicated correlation between samples. The interpolation MDS is used to reduce the dimension of sample so that the clustering results can be visualized in 3D coordination. Through DNA clustering example analysis, the analysis results show that the clustering method is useful. It provides a novel means to solve large scale clustering problems.

Acknowledgements

This work is partially supported by Provincial Outstanding Research Award Fund for young scientist (No. BS2009DX016) and Provincial Fund for Nature project (No. ZR2009FM038).

References

[1] J R Swedlow, G Zanetti, C Best. Channeling the data deluge. Nature Methods, 2011, 8: 463-465.
[2] G C Fox, X H Qiu et al. Case Studies in Data Intensive Computing: Large Scale DNA Sequence Analysis. The Million Sequence Challenge and Biomedical Computing Technical Report, 2009
[3] X H Qiu, J Ekanayake, G C Fox et al. Computational Methods for Large Scale DNA Data Analysis. Microsoft eScience workshop, 2009
[4] J A Blake, C J Bult. Beyond the data deluge: Data integration and bio-ontologies. Journal of Biomedical Informatics, 2006, 39(3), 314-320.
[5] J Qiu. Scalable Programming and Algorithms for Data Intensive Life Science. Applications Data-Intensive Sciences Workshop, 2010
[6] R Guha, K Gilbert, G C Fox, et al. Advances in Cheminformatics Methodologies and Infrastructure to Support the Data Mining of Large, Heterogeneous Chemical Datasets. Current Computer-Aided Drug Design, 2010, 6: 50-67.
[7] C C Chang, B He, Z Zhang. Mining semantics for large scale integration on the web: evidences, insights, and challenges. SIGKDD Explorations, 2004: 6(2):67-76.
[8] G C Fox, S H Bae, et al. Parallel Data Mining from Multicore to Cloudy Grids. High Performance Computing and Grids workshop, 2008
[9] B J Zhang, Y Ruan et al. Applying Twister to Scientific Applications. Proceedings of CloudCom, 2010
[10] J Ekanayake, H Li, et al. Twister: A Runtime for iterative MapReduce. The First International Workshop on MapReduce and its Applications of ACM HPDC, 2010
[11] http://www.public.iastate.edu/~apghosh/files/IEEEclust2.pdf. Ranjan Maitra, Anna D. Peterson and Arka P. Ghosh. A systematic evaluation of different methods for initializing the K-means clustering algorithm. 2010
[12] Haykin, Simon (1999). "9. Self-organizing maps". Neural networks - A comprehensive foundation (2nd ed.). Prentice-Hall
[13] N. Tishby, C. Fernando, W. Bialek, “The information bottleneck method,” The 37th Annual Allerton Conference on Communication, Control and Computing, Monticello, Sep. 1999, pp. 1-11.
[14] J. Coldberger, S. Gordon, H. Greenspan, “Unsupervised image-set clustering using an information theoretic framework,” IEEE transactions on image processing, vol.15, no. 2, pp. 449-457, 2006.
[15] N. Slonim, T. Somerville, N. Tishby, “Objective classification of galaxy spectra using the information bottleneck method,” Monthly Notices of The Royal Astronomical, vol. 323, pp. 270-284, 2001.
[16] Jolliffe, I. T. Principal component analysis. New York : Springer, 2002.
[17] George K Matsopoulos. Self-Organizing Maps. INTECH, 2010.
[18] Borg Ingwer, Patrick J.F. Croenen. Modern Multidimensional Scaling: Theory and Applications. New York : Springer, c2005. pp. 207–212
[19] Seung-Hee Bae, Judy Qiu, Geoffrey Fox. Adaptive Interpolation of Multidimensional Scaling International Conference on Computational Science ICCS Omaha Nebraska June 4-6 2012