J Supercomput (2014) 69:452–467
DOI 10.1007/s11227-014-1174-1

A parallel clustering method combined information bottleneck theory and centroid-based clustering

Zhanquan Sun · Geoffrey Fox · Weidong Gu · Zhao Li

Published online: 4 April 2014

Abstract Clustering is an important research topic of data mining. Information bottleneck theory-based clustering method is suitable for dealing with complicated clustering problems because that its information loss metric can measure arbitrary statistical relationships between samples. It has been widely applied to many kinds of areas. With the development of information technology, the electronic data scale becomes larger and larger. Classical information bottleneck theory-based clustering method is out of work to deal with large-scale dataset because of expensive computational cost. Parallel clustering method based on MapReduce model is the most efficient method to deal with large-scale data-intensive clustering problems. A parallel clustering method based on MapReduce model is developed in this paper. In the method, parallel information bottleneck theory clustering method based on MapReduce is proposed to determine the initial clustering center. An objective method is proposed to determine the final number of clusters automatically. Parallel centroid-based clustering method is proposed to determine the final clustering result. The clustering results are visualized

Z. Sun (✉) · W. Gu
Key Laboratory for Computer Network of Shandong Province,
Shandong Computer Science Center, 19 Keyuan Road, Jinan 250014, Shandong, China
e-mail: sunzhq@sdas.org

W. Gu
e-mail: guwd@sdas.org

G. Fox
School of Informatics and Computing, Pervasive Technology Institute,
Indiana University Bloomington, Bloomington, IN 47408, USA
e-mail: gcf@indiana.edu

Z. Li
School of Software Engineering, Beijing Jiaotong University, Beijing 100044, China
e-mail: liz@sdas.org

with interpolation MDS dimension reduction method. The efficiency of the method is illustrated with a practical DNA clustering example.

Keywords Clustering · Information bottleneck theory · MapReduce · Centroid-based clustering

1 Introduction

Clustering is a main task of explorative data mining, and a common technique for statistical data analysis used in many areas, such as machine learning, pattern recognition, image analysis, information retrieval, bioinformatics and so on. The goal of clustering is to determine the intrinsic grouping in a set of unlabeled data. Some classical clustering methods, such as centroid-based clustering, Fisher clustering method, Kohonen neural network and so on, have been studied and widely applied to many kinds of field [1]. Centroid-based clustering, such as k-mean, k-center and so on, is a kind of important clustering method [2]. It is easy to be used in practice. But it has some shortcomings. The first one is that no objective method is available to determine the initial center, which has great effect on the final clustering results. Another one is that the number of clusters cannot be determined objectively. Furthermore, the distance metric used in centroid-based clustering usually cannot measure the complicated relationships between samples. Information bottleneck (IB) theory was proposed by Tishby et al. [3]. 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. The quantity of information loss caused by merging is used to measure the distance between samples. It has been applied to the clustering of image, texture, and galaxy successfully and got good results [4,5]. However, the computation cost of IB clustering is expensive. It will be out of work to deal with large-scale dataset. With the development of electronic and computer technology, the quantity of electronic data increases exponentially [6]. 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 [7,8], biomedical [9,10], Cheminformatics [11], web [12] and so on. How to develop parallel clustering methods to process large-scale data is an important issue. Many scholars have done lots of work on this topic. Efficient parallel algorithms and implementation techniques are the key to meeting the scalability and performance requirements entailed in such large-scale data mining analysis. 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 [13]. MPI model is efficient in computation intensive problems, especially in simulation. However, it is not efficient in dealing with data-intensive problems. MapReduce is a programming model developed from the data analysis model of the information retrieval field. Several MapReduce architectures are developed, such as Barrier-less MapReduce, MapReduceMerge, Oivos, Kahn process networks and so on [14]. But all these MapReduce architectures do not support iterative Map and Reduce tasks, which is required in many data mining algorithms. An iterative MapReduce architecture software Twister is developed by Fox. It supports not only noniterative MapReduce applications but also iterative MapReduce applications [15]. It can be used in data-intensive data mining problems. 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 cannot be transformed to MapReduce model directly. Furthermore, the number of clusters of IB clustering should be determined manually according to an objective rule. It cannot be operated automatically.

The evaluation of unsupervised clustering result is a difficult problem. Visualization is a good mean to improve it. However, in practical, many problems' feature variable vectors are in high dimensions. Feature extraction can decrease the dimension of input 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]. 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. The computation complexity is $O(n^2)$, and memory requirement is $O(n^2)$. With the increase in sample size, the computation cost of MDS increase sharply. For improving the computation speed, interpolation MDS is introduced in [19]. It is used to extract features from large-scale data. In this paper, interpolation MDS is used to reduce the feature dimension.

In this paper, a novel clustering method based on MapReduce is proposed. It combines parallel IB theory clustering with parallel centroid-based clustering. Firstly, IB theory-based hierarchy clustering is used to determine the centroid of each Map computational node. An objective method is proposed to determine the number of clusters. All subcentroids are combined into one centroid with the IB theory also in Reduce computational node. The centroid is taken as the initial center of centroid-based clustering method. For measuring the complicated correlation between samples, information loss is used to measure the distance in the centroid-based clustering method. The clustering method is programmed with iterative MapReduce model Twister. For visualizing the clustering results, interpolation MDS is used to reduce the samples into 2 or 3 dimensions. The reduced clustering results are shown in 3D coordination with Pviz software developed by Indiana University. A DNA clustering example is analyzed with the proposed method to illustrate the efficiency.

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

2 Parallel IB clustering

2.1 IB principle

The IB clustering method states that among all the possible clusters 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 [3]. Let $p(x, y)$ be a joint distribution 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 $\hat{X}$

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

The loss of the mutual information between X and Y caused by the clustering $\hat{X}$ can be calculated as follows.

$$\begin{aligned} d(x,\hat{x}) = I(X;Y) - I(\hat{X};Y) = \sum_{x,\hat{x},y} p(x,\hat{x},y)\log\frac{p(y|x)}{p(y)} \\ - \sum_{x,\hat{x},y} p(x,\hat{x},y)\log\frac{p(y|\hat{x})}{p(y)} = ED(p(x,\hat{x})||p(y|\hat{x}) \end{aligned} \tag{2}$$

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

$$d(c_1, c_2) = I(c_1; Y) + I(c_2; \mathrm{Y}) - \mathrm{I}(c_1, c_2; Y) \tag{3}$$

Standard information theory operation reveals

$$d(c_1, c_2) = \sum_{y,i=1,2} p(c_i)p(y|c_i)\log\frac{p(y|c_i)}{p(y|c_1 \cup c_2)} \tag{4}$$

where $p(c_i) = |c_i| / |X|$, $|c_i|$ denotes the cardinality of c_i, $|X|$ denotes the cardinality of object space X,$p(c_1 \cup c_2) = |c_1 \cup c_2| / |X|$, and $p(y|c_i)$ is the probability density of Y in cluster c_i.

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

$$p(y|c_1 \cup c_2) = \sum_{i=1,2} \frac{|c_i|}{|c_1 \cup c_2|} p(y|c_i) \tag{5}$$

The minimization problem can be approximated with a greedy algorithm. The algorithm is based on a bottom-up merging procedure and starts with the trivial clustering where each cluster consists of a single data vector. In each step, the two clusters with minimum information loss are merged. The method is suitable to both sample clustering and feature clustering.

2.2 Determine the number of clusters

The number of final clusters usually is prescribed subjectively in many clustering methods. For avoiding the subjectivity, IB theory-based clustering method provides an objective rule to determine it. The clustering process of IB is iterative, and each step has an information loss value. The number of clusters corresponding to the iterative step whose information loss changes markedly is taken as the final number of clusters. Although the determination rule in IB theory-based clustering is objective, the judgment of information loss change is done manually. It is inconvenient to be operated in parallel clustering. An objective judgment method is proposed to determine the final step whose information loss changes markedly. The method is described as follows.

Suppose the information loss of previous k steps were known, the information loss value of current step is estimated with least square regression method. The clustering procedure will stop when the difference between estimated and practical information loss value is greater than a threshold value β whose value range is [0–1]. The value β can be prescribed according to practical problems.

1. Least square regression

Linear regression finds the straight line that best represents observations in a bivariate dataset. Suppose Y is a dependent variable, and X is an independent variable. The regression line is

$$y = ax + b \tag{6}$$

where b is a constant, a is the regression coefficient, x is the value of the independent variable, and y is the value of the dependent variable. Given a random sample of observations, the population regression line is estimated by

$$\min \sum_{i=1}^{k-1} (y_i - (ax_i + b))^2 \tag{7}$$

After introducing the Lagrange coefficient, the optimum solution of the equation is

$$\hat{a} = \frac{\sum_{i=1}^{k-1} x_i y_i - \left(\sum_{i=1}^{k-1} x_i \sum_{i=1}^{k-1} y_i\right)/m}{\sum_{i=1}^{k-1} x_i^2 - \left(\sum_{i=1}^{k-1} x_i\right)^2/m} \tag{8}$$

$$\hat{b} = \frac{\sum_{i=1}^{k-1} y_i}{m} - \hat{a}\frac{\sum_{i=1}^{k-1} x_i}{m} \tag{9}$$

According to the optimum parameter â and b̂, the estimated information loss of current step is

$$\hat{y}_i = \hat{a}x_i + \hat{b} \tag{10}$$

2. Determination of the number of clusters

In the regression, clustering step is taken as X and each step's information loss value is taken as Y. The difference between estimated value $\hat{y}_i$ and the practical information loss value y_i is measured with the following equation.

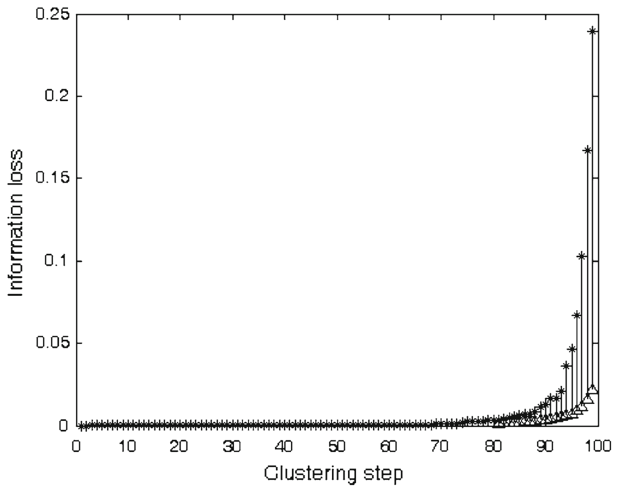

Fig. 1 The clustering procedure based on information bottle-neck theory

$$e = \frac{y_i - \hat{y}_i}{y_i} \tag{11}$$

Clustering procedure will stop when $e > \beta$. A clustering example based on the procedure can be shown as in Fig. 1. There are 100 samples in total. The threshold value of β is set 0.9. The X axis denotes the clustering step, and Y axis denotes the information loss value. After calculation, the difference value e of step 94 is greater than the threshold value. Then, we can obtain the clustering number 6 automatically.

2.3 Parallel IB based on MapReduce

Given a dataset D with n samples, it is divided into m partitions $D^1, D^2, \ldots, D^m$ with $n_1, n_2, \ldots, n_m$ samples separately. Apply the clustering method introduced as above to each partition $D^i = \{D_1^i, D_2^i, \ldots, D_{n_i}^i\}, i = 1, \ldots, m$. We can obtain the subcentroids $C^i, i = 1, \ldots, m$. All subcentroids are collected together to generate new data set $C = \{C^1, C^2, \ldots, C^m\}$. After applying the proposed clustering method to the new dataset, we can obtain the initial global center C^0 and the number of clusters k. From Eq. (5), the cardinality of each cluster is required. The sample size of each subcentroid should be saved so that they can be used to calculate the final clustering result. The parallel calculation process based on MapReduce is shown in Fig. 2. Firstly, partitioned datasets are deployed to each computational node evenly. In each Map computational node, apply IB theory-based clustering method to each subdataset to obtain the subcentroid. All subcentroids are collected in Reduce node to generate a new dataset. Apply IB theory-based clustering method to the new dataset to generate the initial centroid of the global dataset.

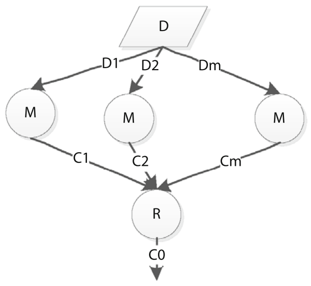

Fig. 2 The calculation process of parallel IB based on MapReduce

3 Parallel centroid clustering based on iterative MapReduce

After the initial center C^0 being calculated, it is used to calculate the final centroid. The process is as follows. Firstly, calculate the distance between each sample $\boldsymbol{x}$, $\boldsymbol{x} \in D^i$, and all the centers of the centroids C^0. In the calculation, information loss (4) is taken as the distance measure. Let $P^1, P^2, \ldots, P^k$ be k empty dataset. The sample x will be added to dataset P^i if the distance between x and center vector c_i^0 is the minimum. Recalculate the centroids C^j of computational node j, $j = 1, 2, \ldots, m$ with the datasets $P^1, P^2, \ldots, P^k$ according to (5). After calculating the new subcentroids $C^1, C^2, \ldots, C^m$, the update centroid C^0 can be calculated according to the following equation.

$$c_i^0 = \sum_{j=1}^{k} \frac{\left|C^j\right|}{\left|C^1 \cup C^2 \cdots C^k\right|} c_i^j \tag{12}$$

The iteration procedure will stop when the difference δ between the old centroids C^{old} and the new generated centroids C^{new} is less than the threshold value ε. The difference δ between two iterations is measured with Kull-back divergence, i.e.,

$$\delta = \sum_{i=1}^{l} x_i^{new} \log \frac{x_i^{new}}{x_i^{old}} + \sum_{i=1}^{l} x_i^{old} \log \frac{x_i^{old}}{x_i^{new}} \tag{13}$$

The iteration process of parallel centroid clustering based on MapReduce is shown as in Fig. 3. Firstly, initial sample dataset is partitioned and deployed to each compu-

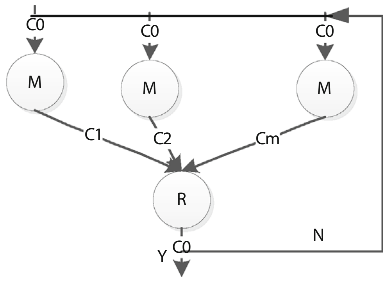

Fig. 3 The calculating process of parallel centroid-based clustering method

tational node. The initial centroids C^0 obtained with parallel IB are mapped to each computational node. In each Map computational node, the subcentroids are recalculated with centroid-based clustering method introduced as above. All subcentroids are collected in Reduce computational node, and the global centroid C^0 is updated according to (12). The new centroids are feedback to main computational node, and the difference δ is calculated according to (13). Iteration will stop when the difference is less than the prescribed threshold value ε.

4 Dimension reduction method

To visualize the clustering results, high-dimensional samples should be mapped into 2 or 3 dimensions. MDS is an efficient dimension reduction method. It is as follows [19].

4.1 Multidimensional scaling

MDS is a nonlinear 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 $D = \{x_1, x_2, \ldots, x_n\}, x_i \in R^N (i = 1, 2, \ldots, n)$ on which a distance function is defined as $\delta_{i,j}$, the pairwise distance matrix of the n objects can be denoted by

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

where $\delta_{i,j}$ 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 $p_1, \ldots, p_n$, $p_i \in R^L (L \leq N)$ to minimize the STRESS or SSTRESS. The definition of STRESS and SSTRESS is as follows.

$$\sigma(P) = \sum_{i<j} w_{i,j}(d_{i,j}(P) - \delta_{i,j})^2 \tag{14}$$

$$\sigma^2(P) = \sum_{i<j} w_{i,j}((d_{i,j}(P))^2 - \delta_{i,j}^2)^2 \tag{15}$$

where $1 \leq \mathrm{i} < j \leq n$, $w_{i,j}$ is a weight value $(w_{i,j} > 0)$, $d_{i,j}(P)$ is a Euclidean distance between mapping results of p_i and 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 MDS

One of the main limitations of most MDS applications is that it requires $O(n^2)$ memory as well as $O(\mathrm{n}^2)$ 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 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}, x_i \in R^N (i = 1, 2, \ldots, n)$ in N dimension space, m samples $D_{sel} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ are selected to be mapped into L dimension space $P_{sel} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m\}$ with MDS. The other samples $D_{rest} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n-m}\}$ will be mapped into L dimension space $P_{rest} = \{p_1, p_2, \ldots, p_{n-m}\}$ with interpolation method.

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

$$\sigma(X) = \sum_{i<j} (d_{i,j}(P) - \delta_{i,j})^2 = C + \sum_{i=1}^{k} d_{ip}^2 - 2 \sum_{i=1}^{k} d_{ip} \delta_{ix} \tag{16}$$

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

$$x^{[t]} = \overline{\boldsymbol{p}} + \frac{1}{k} \sum_{i=1}^{k} \frac{\delta_{ix}}{d_{iz}} \left(x^{[t-1]} - \boldsymbol{p}_i \right) \tag{17}$$

where $d_{iz} = \|\boldsymbol{p}_i - x^{[t-1]}\|$ 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 ε. The difference between two iterations is

$$\omega = \frac{x^{[t]} - x^{[t-1]}}{x^{[t]}} \tag{18}$$

5 Example: DNA sequence clustering

5.1 Data source

Dr. Mina Rho in Indiana University provided some 16S rRNA data that can be downloaded from http://salsahpc.indiana.edu/millionseq/mina/16SrRNA_index.html. 100,000 DNA data are selected to be used clustering analysis. DNA sequences are usually denoted by four letters, i.e., A, C, G, T [20]. A DNA sequence can be taken as a nonempty string S of letter set $\sum \left(\sum = \{A, C, T, G\}\right)$, i.e., $S = (s_1, s_2, \ldots, s_n)$, where $n = |S| > 0$ denotes the length of the string. A DNA can be expressed with the frequency character of four 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 [21]

$$f_{s_i, s_j} = \frac{S_i S_j}{|S| - 1} \tag{19}$$

where $s_i, s_j \epsilon \sum$, $S_i S_j$ denotes the frequency of some double sequence nucleic acid in a DNA string. $|S|$ denotes the length of the DNA sequence. In the above formula, the nucleic acids except 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

$$p_{s_i, s_j} = \frac{f_{s_i, s_j}}{f_{s_i} f_{s_j}} \tag{20}$$

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

$$p^*_{s_i, s_j} = \frac{p_{s_i, s_j}}{\sum p_{s_i, s_j}} \tag{21}$$

The sample strings are transformed into 16 dimensions vector. They are described with probabilities and taken as the initial clustering dataset.

The example is analyzed in India cluster node of FutureGrid. Eucalyptus platform is adopted to configure the MapReduce computation environment. Twister0.9 software is deployed in each computational node. ActiveMQ is used as message broker. The configuration of each virtual machine is as follows. Each node installs Ubuntu Linux OS. The processor is 3GHz Intel Xeon with 10GB RAM.

5.2 DNA sequence clustering

The initial sequence dataset is partitioned into 100 sections, and each section includes 1,000 samples. They are deployed to each computational node evenly. Apply parallel IB theory-based clustering to each section. The parameters are set as $\beta = 0.97, \varepsilon = 0.1$ and $\varepsilon = 0.01$. Reduce computational node is used to combine all the subcentroids into one centroid. We got the initial centroids C^0, and the clustering number is determined as 6.

Centroids C^0 are mapped to each computational node. Recalculate the centroid of each partition according to the Sect. 4.2 iteratively. The difference value δ reaches the threshold value ε after five iterations. We can obtain the final centroids C.

Table 1 computation time of parallel IB based on 100 partitions

Node number	1	2	4	8
Computation time(s) based on 100 partitions	3,256	1,742	883	441

Table 2 computation time of parallel IB based on 50 partitions

Node number	1	2	4	8
Computation time(s)	16,018	8,132	4,174	2,201

C = {(0.0610 0.0701 0.0701 0.0486 0.0705 0.0554 0.0663 0.0617 0.0452 0.0729 0.0661 0.0619 0.0677 0.0573 0.0480 0.0762); (0.0597 0.0525 0.0732 0.0670 0.0670 0.0666 0.0534 0.0617 0.0541 0.0615 0.0695 0.0640 0.0661 0.0648 0.0595 0.0586); (0.0514 0.0602 0.0828 0.0568 0.0704 0.0610 0.0654 0.0559 0.0534 0.0642 0.0539 0.0746 0.0698 0.0636 0.0514 0.0643); (0.0662 0.0579 0.0802 0.0499 0.0726 0.0649 0.0599 0.0529 0.0384 0.0648 0.0666 0.0750 0.0658 0.0603 0.0485 0.0752); (0.0596 0.0764 0.0579 0.0532 0.0699 0.0619 0.0574 0.0617 0.0643 0.0541 0.0641 0.0690 0.0529 0.0591 0.0718 0.0661); (0.0616 0.0656 0.0806 0.0459 0.0711 0.0559 0.0608 0.0648 0.0457 0.0616 0.0565 0.0803 0.0665 0.0672 0.0561 0.0590)}

For comparison, the 100 sections are deployed to 1, 2, 4 and 8 computational nodes, respectively. The computation times are listed in Table 1.

From Table 1, we can find that the computation time decreases markedly when the number of computational node increases. It shows that parallel clustering method based on MapReduce is scalable. For illustrating the affection of different partition scheme, the initial dataset are portioned into 50 sections. When the dataset is partitioned into 50 sections, each section includes 2,000 samples. We got the final centroids C. When the dataset is not partitioned, the clustering cannot be operated because of RAM limitation.

C = {(0.0611 0.0720 0.0719 0.0447 0.0721 0.0561 0.0649 0.0608 0.0446 0.0702 0.0668 0.0643 0.0665 0.0569 0.0475 0.0787); (0.0560 0.0584 0.0813 0.0591 0.0610 0.0574 0.0647 0.0685 0.0392 0.0784 0.0684 0.0584 0.0864 0.0610 0.0390 0.0619); (0.0647 0.0633 0.0593 0.0620 0.0685 0.0551 0.0722 0.0588 0.0478 0.0794 0.0631 0.0570 0.0644 0.0585 0.0538 0.0714); (0.0566 0.0600 0.0820 0.0539 0.0711 0.0613 0.0634 0.0563 0.0487 0.0640 0.0569 0.0759 0.0681 0.0634 0.0517 0.0660); (0.0601 0.0525 0.0732 0.0667 0.0673 0.0668 0.0531 0.0613 0.0540 0.0614 0.0694 0.0644 0.0655 0.0646 0.0599 0.0591); (0.0596 0.0764 0.0580 0.0530 0.0699 0.0618 0.0574 0.0618 0.0643 0.0540 0.0640 0.0691 0.0529 0.0592 0.0717 0.0660)}

The computation times based on 1, 2, 4 and 8 computational nodes are listed in Table 2.

From Tables 1 and 2, we can find that computation cost increases markedly when the size of each partition increases. It shows that the parallel clustering method based on MapReduce is efficient in decreasing computation cost.

5.3 Visualization of clustering result

The feature dimension of the initial dataset is 16. For visualizing the clustering result, the initial dataset is mapped into 2D and 3D with interpolation MDS, respectively. In this example, 4,000 samples are selected and mapped into 2D and 3D space with

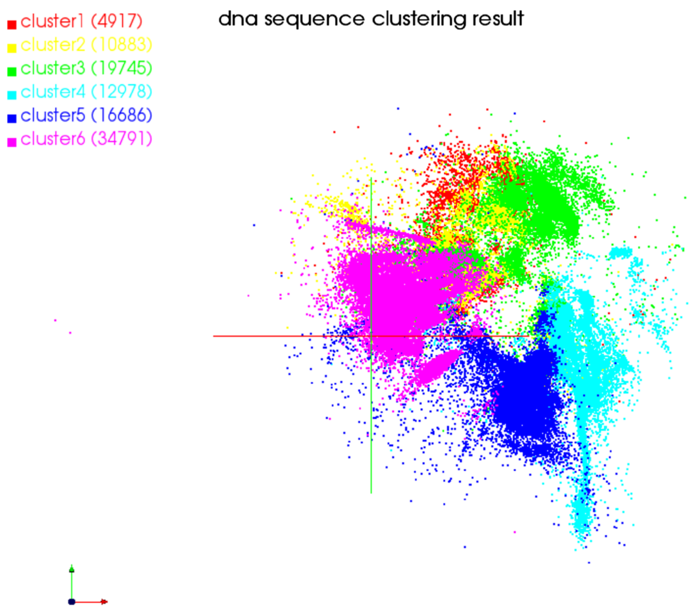

Fig. 4 2D clustering results based on combination of information bottleneck theory and interpolation MDS corresponding to 100 partitions

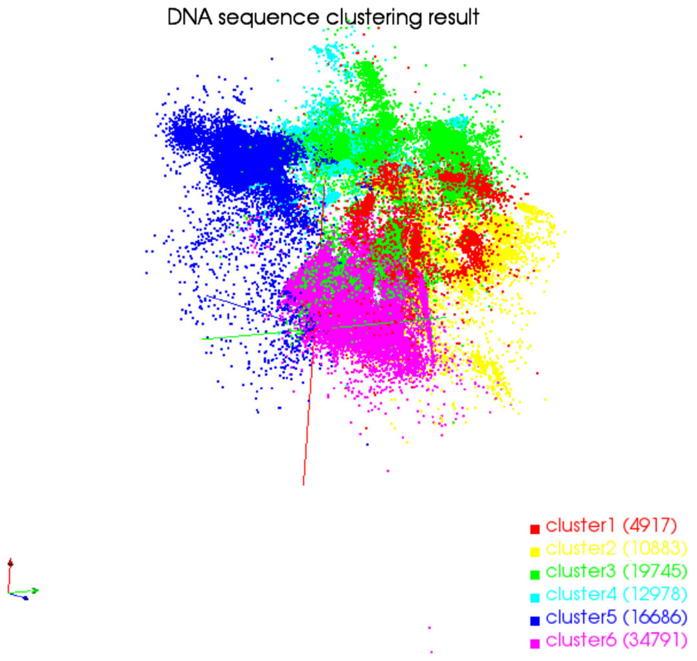

Fig. 5 3D clustering results based on combination of information bottleneck theory and interpolation MDS corresponding to 100 partitions

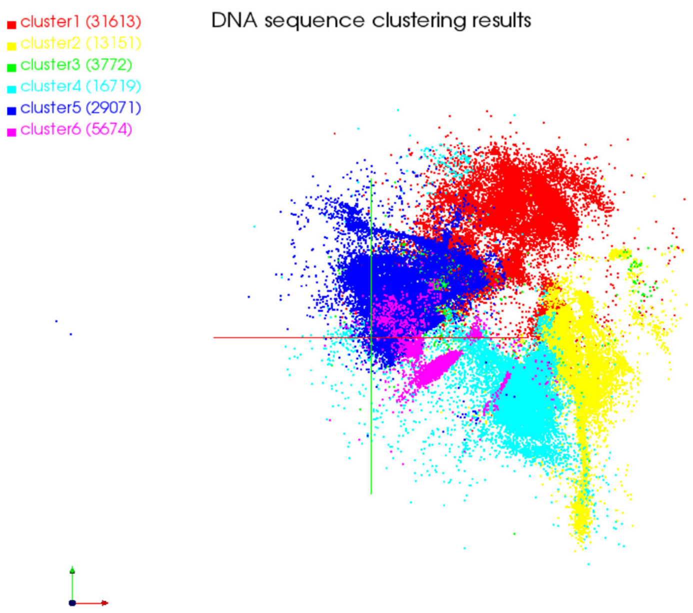

Fig. 6 2D clustering results based on combination of information bottleneck theory and interpolation MDS corresponding to 50 partitions

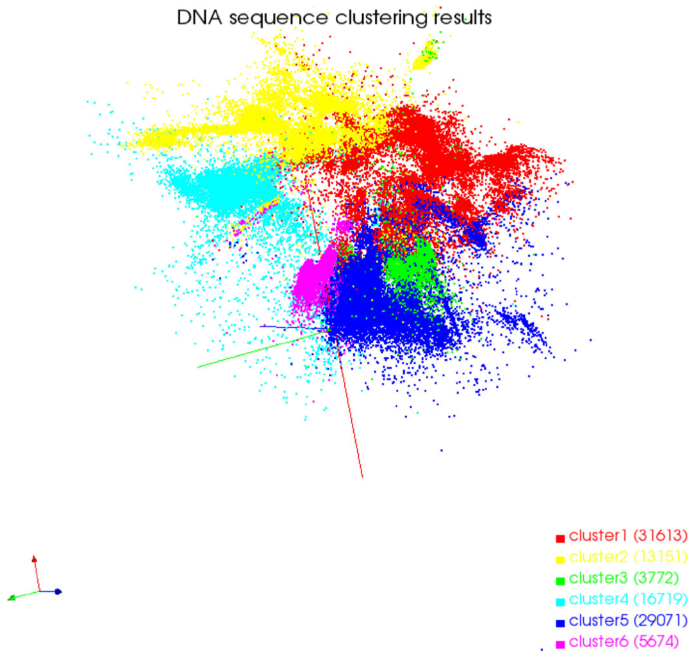

Fig. 7 3D clustering results based on combination of information bottleneck theory and interpolation MDS corresponding to 50 partitions

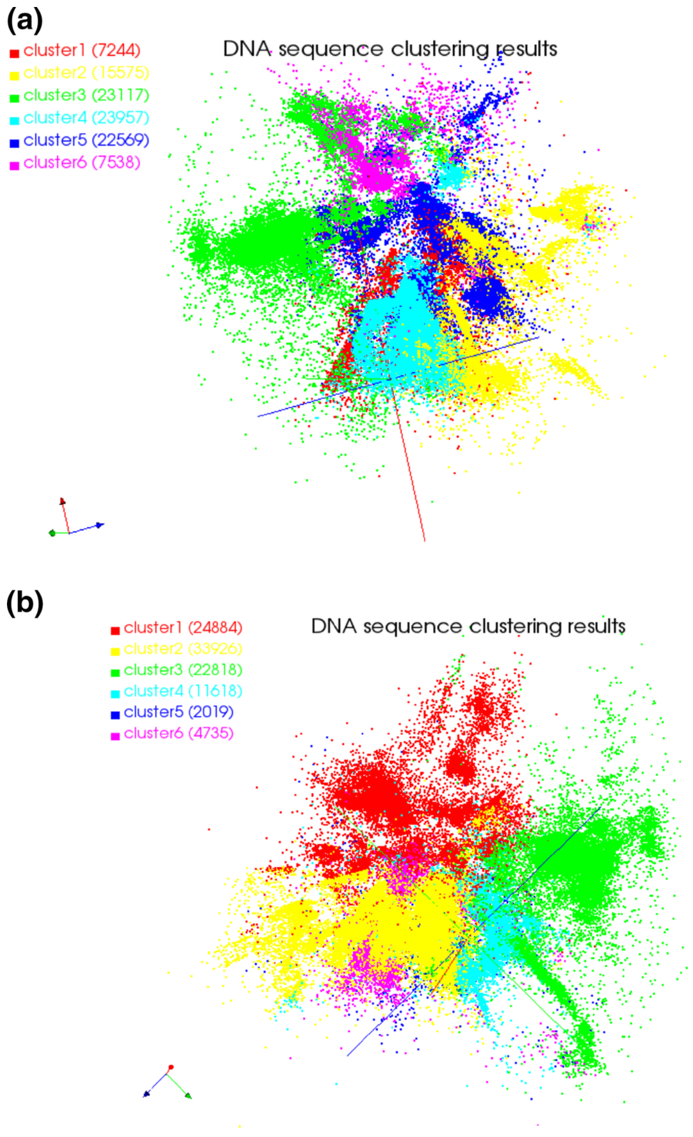

Fig. 8 3D clustering results based on Kmeans with different initial centroids

MDS method. The distance matrix of the 4,000 samples is calculated firstly according to Euclidean distance. Other samples are mapped into 2D and 3D with interpolation MDS method. In the calculation, the number of nearest neighbor is set $k = 10$. After dimension reduction, the clustering results are visualized with the dimension reduction results. The clustering results of 100 partitions are shown in 2D and 3D as in Figs. 4 and 5, respectively.

The clustering results of 50 partitions are shown in 2D and 3D are shown as in Figs. 6 and 7.

5.4 Clustering based on Kmeans

For comparing the clustering results, parallel Kmeans based on MapReduce is used to analyze the example. Dataset is partitioned into 50 sections. The clustering number is set to 6, and the initial centroids are selected from the dataset randomly. The clustering results based on different initial centroid are different. Figure 8a, b is the clustering results based on Kmeans in 3D with different initial centroids.

From above visualization results, we can find that the clustering result based on the proposed method in this paper is better than that of parallel Kmeans method.

6 Conclusions

Large-scale data clustering is an important task in many application areas. Efficient clustering method can reduce the computation cost markedly. The proposed clustering method in this paper is efficient for large-scale data analysis. It is based on MapReduce program model. It can increase the computation speed through increase partition number. On the other hand, the initial clustering centroid and the number of clusters can be determined according to an objective rule automatically. The DNA example analysis results show that the proposed method is scalable. The information loss is used to measure the distance between samples. It can measure any complicated statistical correlation between samples. Interpolation MDS is used to reduce the feature dimension of samples so that the clustering results can be visualized in 2D and 3D. The visualization clustering results of the example show that the clustering result of the proposed method is better than that of Kmeans. Information loss based on mutual information can measure arbitrary statistic correlations. It provides a novel means to solve large-scale clustering problems.

Acknowledgments This work is partially supported by national youth science foundation (No. 61004115), national science foundation (No. 61272433) and Provincial Fund for Nature project (No. ZR2010FQ018).

References

1. Khana SS, Ahmad A (2013) Cluster center initialization algorithm for K-modes clustering. Expert Sys Appl 40(18):7444–7456
2. Sim K, Yap GE, Hardoon DR et al (2013) Centroid-based actionable 3D subspace clustering. IEEE Trans Knowl Data Eng 25(6):1213–1226
3. Tishby N, Fernando C, Bialek W (1999) The information bottleneck method. In: The 37th annual allerton conference on communication, control and computing, Monticello, pp 1–11
4. Coldberger J, Gordon S, Greenspan H (2006) Unsupervised image-set clustering using an information theoretic framework. IEEE Trans Image Process 15(2):449–457
5. Slonim N, Somerville T, Tishby N (2001) Objective classification of galaxy spectra using the information bottleneck method. Mon Not R Astron 323:270–284
6. Swedlow JR, Zanetti G, Best C (2011) Nat. Methods. Channeling the data deluge 8:463–465
7. Fox GC, Qiu XH et al (2009) Biomedical case studies in data intensive computing. Lect Notes Comput Sci 5931:2–18
8. Sun ZQ, Fox GC (2012) Study on parallel SVM based on MapReduce. In: International conference on parallel and distributed processing techniques and applications, CSREA Press, pp 495–501

9. Blake JA, Bult CJ (2006) Beyond the data deluge: data integration and bio-ontologies. J Biomed Inform 39(3):314–320
10. Qiu J (2010) Scalable programming and algorithms for data intensive life science. J Integr Biol 15(4):1–3
11. Guha R, Gilbert K, Fox GC et al (2010) Advances in cheminformatics methodologies and infrastructure to support the data mining of large, heterogeneous chemical datasets. Curr Comput-Aided Drug Des 6:50–67
12. Chang CC, He B, Zhang Z (2004) Mining semantics for large scale integration on the web: evidences, insights, and challenges. SIGKDD Explor 6(2):67–76
13. Fox GC, Bae SH et al (2008) Parallel data mining from multicore to cloudy grids. High performance computing and grids workshop, IOS Press, pp 311–340
14. Li JJ, Cui J, Wang D et al (2011) Survey of MapReduce parallel programming model. Acta Electronica Sinica 39(11):2635–2642
15. Ekanayake J, Li H et al (2010) Twister: a runtime for iterative MapReduce. In: The first international workshop on MapReduce and its applications of ACM HPDC, ACM press, pp 810–818
16. Jolliffe IT (2002) Principal component analysis. Springer, New York
17. George KM (2010) Self-organizing maps. INTECH
18. Borg I, Patrick JF (2005) Modern multidimensional scaling: theory and applications. Springer, New York
19. Bae S-H, Qiu J, Fox G (2012) Adaptive interpolation of multidimensional scaling. In: International conference on computational science, pp 393–402
20. Ananstassiou D (2000) Frequency-domain analysis of biomolecular sequences. Bioinformatics 16(12):1073–1081
21. Liang B, Chen DY (2010) DNA sequence classification based on ant colony optimization clustering algorithm. Comput Eng Appl 46(25):124–126