A New Approach for Large-Scale Scene Image Retrieval Based on Improved Parallel $k$-Means Algorithm in MapReduce Environment

Jianfang Cao,1 Min Wang,2 Hao Shi,2 Guohua Hu,1 and Yun Tian1

1Department of Computer Science & Technology, Xinzhou Teachers University, Xinzhou 034000, China
2School of Computer Science & Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China

Correspondence should be addressed to Jianfang Cao; caojianfangcn@163.com

Received 8 May 2016; Revised 21 July 2016; Accepted 11 August 2016

Academic Editor: Anna Vila

The rapid growth of digital images has caused the traditional image retrieval technology to be faced with new challenge. In this paper we introduce a new approach for large-scale scene image retrieval to solve the problems of massive image processing using traditional image retrieval methods. First, we improved traditional $k$-Means clustering algorithm, which optimized the selection of the initial cluster centers and iteration procedure. Second, we presented a parallel design and realization method for improved $k$-Means algorithm applied to feature clustering of scene images. Finally, a storage and retrieval scheme for large-scale scene images was put forward using the large storage capacity and powerful parallel computing ability of the Hadoop distributed platform. The experimental results demonstrated that the proposed method achieved good performance. Compared with the traditional algorithms with single node architecture and parallel $k$-Means algorithm, the proposed method has obvious advantages for use in large-scale scene image data retrieval in terms of retrieval accuracy, retrieval time overhead, and computational performance (speedup and efficiency, sizeup, and scaleup), which is a significant improvement from applying parallel processing to intelligent algorithms with large-scale datasets.

1. Introduction

Image retrieval is to find the images with the specified feature or the specified content according to the description of the image content in the image set. Due to the complexity of image content and the subjectivity of human cognition, establishing efficient and universal image retrieval has been a very difficult task. Therefore, image retrieval has been the research hotspot in the computer vision and information retrieval fields in recent years [1]. Since the late 1990s, aiming at frame images in videos, Doulamis et al. [2, 3] designed an automatic extraction framework of characteristic frames or scenes for a video sequence and proposed a fuzzy representation of visual content which was useful for content-based image retrieval. Avrithis et al. [4] also presented a content-based indexing and retrieval framework for videos. These works have laid a good foundation for the research of image retrieval. In order to obtain more satisfactory retrieval results, Yang et al. [5] proposed a new semisupervised algorithm and a semisupervised long-term relevance feedback algorithm to design a multimedia retrieval framework and implemented cross-media retrieval, image retrieval, and 3D motion/pose data retrieval. Subrahmanym et al. [6] introduced a new descriptor called local maximum edge binary patterns to represent the local region of images for image retrieval, which was applied to object tracking. Through experiments on four different small-scale datasets, the retrieval accuracy was improved compared with other existing algorithms. Color is one of the basic features of images, which is one of the most widely used visual features by image retrieval. Liu and Yang [7] proposed the color difference histograms to represent color feature of the image. It was different from the existing histogram techniques, which not only counted the number or frequency of pixels but also counted the perceptually uniform color difference between two points under different backgrounds concerning colors and edge orientations in $L^*a^*b^*$.
color space. The results of the experiment confirmed strong discriminative power. Aiming at images of social media websites, Gao et al. [8] put forward a social image search method based on visual-textual joint relevance learning, which used visual and textual information simultaneously to estimate the relevance of user tagged images and utilized a dataset including 370+ images to validate the effectiveness of the proposed approach. Zheng et al. [9] studied image retrieval methods using statistics projection algorithm and Robert algorithm. To retrieve the most relevant images optimizing the time complexity, Madhusudhanarao et al. [10] constructed a model for image retrieval using fusion and relevancy methodology, which integrated the features corresponding to multiple modalities and feature level fusion technique. The performance of the proposed model was evaluated using brain web data of UCI database. Ashraf et al. [11] introduced a new image representation technology, namely, Bandelets transform, used Support Vector Machine for image retrieval, and evaluated the performance on three standard datasets. Ramana et al. [12] analyzed different concepts used to improve the image retrieval efficiency and presented a kind of scope that could improve the performance issues in image retrievals.

However, the production of digital images has rapidly increased with advances in multimedia technology and network technology. Faced with massive amounts of image data, determining how to retrieve the images that users require rapidly and precisely has become the focus of considerable efforts. The abovementioned traditional image retrieval methods based on single-node architecture can largely meet user requirements regarding access time when the number of users who access image database simultaneously is low. With the rapid growth of the number of images, the image feature database has become extremely large, and the number of users who access online image databases simultaneously has also increased, thereby resulting in a rapid decline in retrieval speed. Moreover, because the calculation of image feature similarity is a complex operation, a long computing time and a large amount of computing resources will be consumed when using traditional image retrieval methods. These problems cause the running efficiency of the system to decrease rapidly, even if the response to the user must be performed in a timely manner, when a large number of users attempt to retrieve images from the database simultaneously [13]. Therefore, the traditional image retrieval methods have been unable to complete the processing of large-scale images and cannot easily satisfy the people’s requirements on retrieval performance. Therefore, the technology of retrieving large-scale images faces a significant challenge and exploring new methods to retrieve images has become a popular research topic in the digital imaging field. The development of cloud computing technology provides a new concept for the processing of large-scale images. Cloud computing technology has a close relationship with big data. Therefore, using a cloud computing platform to enable distributed parallel processing is an effective solution for realizing the efficient retrieval of large-scale images. Hadoop, which is a software framework of distributed parallel processing for all types of large-scale data, has been widely applied in numerous fields due to its excellent large-scale data processing ability, good extensibility, high reliability, and low cost [14]. In recent years, researchers have begun to focus on large-scale image processing. Almeer [13] realized the analysis and processing of massive remote sensing images using the Hadoop Distributed File System (HDFS) of Hadoop. From the perspective of mass image processing technology, Wiley et al. [15] realized the analysis and processing of astronomical images using Hadoop through converting images into serialized binary files. However, the application is subject to certain restrictions due to its inability to read and write images randomly. Zhu [16] proposed an image classification method by defining an image interface to read and write the entire image. However, the method does not consider the problem of low efficiency for small documents, which causes resource waste; thus, it is only suitable for processing remote sensing images. Sweeney et al. [17] proposed a new method that converted image data into a float array and stored the array in a file with an index file. The method effectively solved the problem of low efficiency for small documents and supported random access through the index file; however, the method is limited to certain fields because the method is unable to store the original complete information of images and only supports RGB color space and images with JPEG, PNG, and ppm format and the method of coding and decoding conversion between image and array is more complex. Scholars also put forward some methods to deal with different types of large-scale images. Chen et al. [18] proposed a new approach to estimate a set of possible ages according to a facial image for large image database. Considering landmark image search, Cheng and Shen [19] developed a very large-scale test collection to support robust performance evaluation. Makantasis et al. [20] realized retrieval for 31,000 cultural heritage images by exploiting and fusing two unsupervised clustering techniques: DBSCAN and spectral clustering. There are no relevant literatures on large-scale scene image processing currently.

Clustering is a common data analysis method under the unsupervised learning environment, the purpose of which is to divide the data on the basis of the degree of mutual dependence between data and further help users correctly analyze and extract the potential rules and patterns existing in the data [21]. k-Means algorithm [22] is a popular clustering method and commonly used for feature clustering in image retrieval. But its performance heavily depends on the initial starting conditions [23]. In view of this, researchers put forward a lot of improved K-Means algorithms. Pelleg and Moore [24] proposed x-Means algorithm, Likas et al. [25] presented the global k-Means algorithm, and Arthur and Vassilvitskii [26] brought forward k-Means++ algorithm. In image retrieval field, Górecki et al. [27] investigated the problem of visually similar image retrieval, proposed a novel K-Means Voting algorithm, and obtained more accurate results compared with a classical similarity measure based on the Euclidean metric. Aiming at extremely high-dimensional and sparse image vectors, Cao et al. [28] presented a summation-based incremental learning algorithm for Info-k-Means clustering for image indexing. Belhauari et al. [29] proposed optimized k-Means algorithm which could...
find the optimal centers for each cluster to recognize human face. Younus et al. [30] integrated PSO algorithm and \( k \)-Means clustering algorithm to realize content-based image retrieval, which used PSO algorithm to optimize initial cluster center of \( k \)-Means algorithm. However, the improvement of the above traditional \( k \)-Means clustering algorithm mainly focuses on the determination of the initial cluster centers or distance function. With the rapid increase of the image number, the time efficiency of the above algorithms would drop dramatically. In recent years, scholars have performed research studies on the parallel \( k \)-Means clustering algorithm. The Hadoop distributed platform, which uses the MapReduce parallel programming model to realize storage and calculation of large-scale data, is widely applied in current research fields. Zhao et al. [31] proposed a parallel \( k \)-Means clustering algorithm based on the cloud platform using the Hadoop distributed platform, which performed local combination using the combine function of the MapReduce parallel programming model and increased the iteration speed of the algorithm. Jin and Wang [32] reduced repeated communication traffic and increased the data transfer rate through the addition of a communication module in the computational model. For the sake of Synthetic Aperture Radar (SAR) image change detection and real-time demand, Zhu et al. [33] proposed a parallel fast global \( k \)-Means algorithm, which parallelized the selection of initial cluster centers. Experiments got a good speedup ratio. Although these algorithms can process large-scale data, the clustering quality of these algorithms is not high, and they do not effectively address the problem of the large numbers of calculations in the process of performing one of these algorithms.

In summary, no suitable, effective feature clustering method and retrieval method exists for large-scale scene images currently. Therefore, based on the above studies and taking the traditional \( k \)-Means algorithm and large-scale scene images as research objects, this study considers the feature clustering optimization method using improved parallel \( k \)-Means algorithm for scene images in MapReduce environment and provides an analysis of how to retrieve the user needs of images from the massive scene image database in an accurate, rapid, and efficient manner and realize “people-oriented” efficient retrieval. Compared with the traditional image retrieval methods, the originality of this paper is reflected in the following two aspects. (1) This paper designs distributed parallel \( k \)-Means clustering algorithm and realizes the parallel feature clustering for large-scale scene images using MapReduce parallel programming model. In the process of parallel design, we not only apply Canopy algorithm to optimize the initial clustering centers of \( k \)-Means algorithm, but also design the combine function to optimize the iterative process of clustering, which effectively reduces the communication overhead between node computers in the cluster. (2) This paper presents a distributed parallel storage and retrieval scheme for large-scale scene images in Hadoop platform. The validity is verified from different angles (e.g., storage consuming, retrieval accuracy, retrieval time, the system speedup, and efficiency) by several sets of experiments.

2. Theoretical Background

2.1. \( k \)-Means Algorithm. The \( k \)-Means algorithm [34] is a classical clustering algorithm based on distance; this algorithm takes distance as evaluation index of similarity; in other words, objects that are closer to one another are more similar. The main concept behind this algorithm is as follows: first, randomly select \( k \) points as the initial cluster centers; next, calculate the distance between each sample point and central point and then classify the sample points into the nearest cluster; finally, calculate the new cluster centers of the adjusted classes. If the cluster centers of adjacent twice do not change, then do not change the sample; in this case, the sample adjustment ends, and the clustering criterion function \( E \) converges:

\[
E = \sum_{i=1}^{k} \sum_{p \in C_i} \| p - M_i \|^2 ,
\]

where \( M_i \) is the mean of the data objects in class \( C_i \) and \( p \) is the space point of \( C_i \).

The main concept of the parallel \( k \)-Means algorithm [34] is as follows: first, randomly select a site as the main point; second, the main point uses the \( k \)-Means algorithm to divide it into \( k \) clusters; third, the main point broadcasts the central point of every cluster to the remaining \( k-1 \) subsites; finally, for each subsite, the distance between data objects of itself and the central point of each cluster is calculated, and then, the sample points are classified into the nearest central point, with transfer of the sample points that do not belong to the subsite to the corresponding site that is the cluster of the sample object. This process is repeated until the discriminant function \( E \) converges.

However, the performance of \( k \)-Means algorithm depends on the initial centers and the sample discrimination to a large extent. For sample discrimination, the experimental data in this paper is SUN Database which is a scene image database and the feature distinction of different kind of scene image is better. To reduce the influence which is caused by the randomness of initial clustering centers on algorithm performance, we optimize the selection of the initial cluster centers using Canopy algorithm in advance.

2.2. MapReduce Parallel Programming Model. As one of the core technologies of Hadoop distributed processing, MapReduce provides a type of underlying distributed parallel computing mode for processing big data and a set of complete programming interfaces and an execution environment for developers. MapReduce [35] uses a standard programming and computing mode that can take a function called a high-order function as a parameter to transfer and convert the computational process of data into the executable process of function.

MapReduce divides the computational process of data into two stages, Map and Reduce, corresponding to the two functions mapper and reducer, respectively. First, the original data are split into segments and treated as inputs to the mapper function. Next, after being filtered and converted, the original data become middle results, which are
3.1. Parallel Feature Extraction for Scene Images. SURF algorithm [36] can not only maintain independent characteristics in scale, rotation, and illumination change, but also enable the calculation process to be more efficient. It has become the widely used image local feature extraction method in the field of image retrieval due to its robustness and many researchers [37–43] often regard SURF algorithm as the main feature extraction method when processing digital images. Therefore, this paper applies the SURF algorithm to extract the features of the scene images. The process is as follows.

**Step 1.** Calculate the Hessian matrix of each pixel \( X = (x, y) \) of the scene image under the scale \( \sigma \):

\[
H(X, \sigma) = \begin{bmatrix}
L_{xx}(X, \sigma) & L_{xy}(X, \sigma) \\
L_{xy}(X, \sigma) & L_{yy}(X, \sigma)
\end{bmatrix},
\]

where \( L_{xx}(X, \sigma) \) is the convolution between the Gauss second derivative \((\sigma^2 / \partial x^2)g(\sigma)\) and the pixel \( X \) of the scene image. The other elements of the matrix are obtained in a similar manner.

The matrix is composed of second derivatives and is calculated by an approximate Gauss kernel of the different scales \( \sigma \). Therefore, Hessian value is a function containing three variables: \( H(x, y, \sigma) \).

**Step 2.** Calculate the corresponding position and scale that can reach the maximum value in the spatial domain and scale domain simultaneously.

For every feature point, perform the following tasks: calculate the response \( dx \) and \( dy \) of the Haar wavelet for which the radius of circle is \( 6\sigma \) in the direction of \( x \) and \( y \); add the response with the range of 60° and higher; rotate the window and determine the direction of the longest vector (main direction).

Structure a square area of size \( 20\sigma \) according to the obtained main direction; split the square area into \( 4 \times 4 \) small regions; for every small region, select 25 sampling points, calculate the responses \( dx \) and \( dy \) in the directions of \( x \) and \( y \), respectively, and take the sum of these quantities; next, extract the values of the 4 descriptors: \( \sum dx, \sum dy, \sum |dx|, \sum |dy| \); finally, the 64-dimensional feature vector is obtained.

**Step 3.** Normalize the 64-dimensional feature vectors.

The MapReduce parallel computational process of feature extraction on scene images in the Hadoop distributed platform is described as follows.

**Map Task**

Input: (image_id, image_data)  
Output: (image_id, image_feature).
The Mapper function utilizes the SURF algorithm to extract the feature vectors for each scene image and count the feature number to facilitate the normalization process.

Reduce Task. The reducer function regards each output key value of the mapper function as its input and then passes the value to the output section.

3.2. Feature Clustering Using Improved Parallel k-Means Algorithm. The k-Means clustering algorithm is an iterative algorithm, in which each iteration process requires considerable time and traffic in the distributed environment. Moreover, the time complexity of the traditional parallel k-Means algorithm applied in the feature clustering for images increases because of the higher property dimensions and higher quality of the images. This paper describes the improvement of the traditional parallel k-Means algorithm. First, the Canopy algorithm is used to select the initial cluster centers, and then, the Combine function is used to perform local merging in the process of generating clustering centers, which not only optimizes the initial cluster centers but also optimizes the iterative process and greatly reduces the time complexity of the parallel k-Means algorithm.

3.2.1. Parallel Optimization of Initial Clustering Centers for k-Means Based on Canopy Algorithm. Canopy algorithm [34] is a simple, fast, and accurate algorithm for grouping objects into classes. The algorithm divides clustering into two stages: first, a simple and fast method of computing the distance is used to divide the data into overlapping subsets, each of which is called a “canopy”; second, the distances are calculated using a precise and rigorous distance calculating method for all data vectors that belong to the same “canopy” that appeared in the first stage. In this paper, the Canopy algorithm and K-Means algorithm are integrated; the Canopy algorithm is used to optimize the initial clustering centers of the k-Means algorithm. The parallel optimized process for the initial clustering centers using the Canopy algorithm is as follows:

Input: Scene image dataset List (form as ⟨image_id, image_feature⟩).

Output: k initial clustering centers (form as ⟨canopy_id, image_feature⟩).

Step 1 (Data preprocessing). Sort the scene image dataset List according to the image_id of the images, and set the initial distance thresholds T1 and T2 (obtained by cross validation), T1 > T2.

Step 2. The mapper function randomly selects a sample vector of scene image as a central vector of the canopy and then traverses the scene image dataset. If the distance between the scene image data and central vector of the canopy is less than T1, then the image data are classified the canopy; if the distance between the scene image data and central vector of canopy is less than T2, then the image data are removed from the original dataset. Proceed with this processing until List is a null set. Finally, all of the central vectors of the canopy are output.

Step 3. The reducer function processes the output of the mapper function, integrates the central vectors of the canopy in the Map stage, and generates new central vectors of canopy, that is, the initial clustering centers.

Thus, we obtain the k initial clustering centers.

3.2.2. Parallel Optimization of the Iterative Process for Feature Clustering Based on k-Means Algorithm. The iterative optimization process using the k-Means algorithm for feature clustering is as follows:

Input: scene image dataset List (form as ⟨image_id, image_feature⟩).

Output: k clustering centers (form as ⟨image_id, image_feature⟩).

Step 1. The mapper function receives the output of the reducer function of the Canopy algorithm, calculates the distance between every scene image data and the nearest clustering center of canopy, and outputs scene image data and the respective cluster, such as ⟨cluster_id, image_feature⟩.

Step 2. The combine function receives the output of the mapper function and combines the objects belonging to the same cluster locally, which sums the corresponding dimensions of the scene image data in every cluster and then counts the number of data objects, resulting in the outputs, such as ⟨cluster_id, sum, num⟩.

Step 3. The reducer function receives the output of the combine function, takes the sum of corresponding dimensions of all scene images of every cluster, determines the total number of scene image data, obtains new values of the cluster centers as the stable cluster center of k-Means such as ⟨cluster_id, image_feature, canopy_id⟩, and determines whether the k-Means algorithm is convergent.

Step 4. Perform clustering according to the stable cluster centers. The mapper function receives the scene images to be clustered as inputs, loads the stable cluster centers of the k-Means, calculates the distance between every scene image data and k cluster centers, and then determines the final cluster of the scene image data; the reducer function receives the output of the mapper function, performs data collection, and then obtains the final clustering result.

In addition, for the calculation of similarity between scene image features, this paper uses the Euclidean distance formula, which is a widely used distance definition:

\[ d(X, Y) = \left( \sum_{i=1}^{n} (x_i - y_i)^2 \right)^{1/2}. \]  

(3)
4. Implementation of Large-Scale Scene Image Retrieval

4.1. The Overall Architecture of Retrieval System. To solve the bottleneck problem in the traditional single node architecture due to the rapid growth of scene images, we develop a retrieval scheme for large-scale scene images based on the Hadoop distributed platform. The overall system architecture is shown in Figure 2.

The overall system architecture is divided into three layers:

1. **The Presentation Layer.** Users obtain services through the Internet and submit sample images or receive retrieval results.

2. **The Business Logic Layer.** The Web server executes the corresponding business processing tasks according to the users’ retrieval requests.

3. **Data Processing Layer.** This layer is the core of the entire system and is mainly responsible for storage, management, feature extraction, feature matching, and result output for large-scale scene images. Users submit sample images or retrieval keywords to the Hadoop distributed system, which then performs feature extraction (for a sample image) and feature matching. If a sample image is used to retrieve images, then the system uses the sample image to match features in the database stored in the HDFS of scene images; if keywords are used to retrieve images, then the system uses keywords to match the annotation information stored in the scene image database. Finally, the retrieval results are output.

4.2. The Feature Storage Method for Large-Scale Scene Images. As the core subproject of Hadoop, HDFS adopts the master-slave (Master/Slave) mode and stores large-scale data in a plurality of the associated computers, which, in addition to increasing the storage capacity, also realizes automatic fault tolerance, automatically detects and rapidly recovers hardware failures, and conveniently accesses flow data on large-scale dataset. If the images are all stored in HDFS when the image set is large, then reading these images would require a considerable period of time. HBase is a distributed database-oriented column above HDFS and can read and write in real time. Therefore, the storage path and the features of scene images are stored in HBase in this paper. The structure is described in Table 1.

Take the image ID as the primary key of the HBase table and take the source file of the image and the image features and the annotated information as the two-column family of the HBase table. Due to the implementation of the atomic operation, the HBase table places all information of the image in the same row for reading and writing.

4.3. Large-Scale Scene Image Retrieval. The retrieval process is shown in Figure 3.
LARGE-SCALE SCENE IMAGE RETRIEVAL BASED ON THE IMPROVED DISTRIBUTED $k$-MEANS FEATURE CLUSTER ALGORITHM

Large-scale scene image retrieval based on the improved distributed $k$-Means feature cluster algorithm is based on describing images according to the SURF feature of scene images combined with the annotation information of the scene images. The steps are as follows:

1. Store the scene image database and annotation information in the HBase distributed database of HDFS to extract features and cluster and obtain the retrieval results.

2. Extract the image features in a distributed and parallel manner using the SURF algorithm for scene images in the database; cluster the extracted features using the proposed improved $k$-Means algorithm, and store the clustered image data and features in the HDFS.

3. At the image retrieval stage, the Map task receives the user’s retrieval requirement, reads the image feature database, adjusts the retrieval requirement, extracts the features of the sample image, calculates the similarity degree between the image feature to be retrieved and every clustering center in the image database, and takes the calculation results as the intermediate results to the output.

4. The Reduce task receives the output of the Map task and sorts the similarity degree from large to small and outputs the retrieval results, which are scene images with the greatest similarity degree.

5. Results and Discussion

5.1. Experimental Environment and Test Data. In this paper, the experimental environment adopts the Hadoop cluster consisting of 5 computers (1 computer as the Master node and the remaining 4 computers as the Slave nodes). The basic configuration of every node computer is as follows: 4 G dual...
core processor, 500 GB of hard disk space, and the Ubuntu operating system.

The experimental data are obtained from the COREL Database and the SUN Database on the Internet. The COREL Database published by COREL Corporation consists of 60,000 images (about 100 images represent one image category, that is to say, 600 image categories altogether) based on 600 CD-ROMs. We download freely a subset of 60,000 images containing 10,000 images and 100 categories from http://wang.ist.psu.edu/docs/related/. To construct the larger datasets, we duplicate the 10,000 images (about 100 images per category, 100 categories altogether) to 30,000 images (about 300 images per category, 100 categories altogether) so as to satisfy the experimental requirement. We construct 8 datasets by random selection according to image category and image number and name them as “Data 1” through “Data 8.” The category number and image number included in these datasets are, respectively, as follows: 10 categories, 1,500 images; 10 categories, 3,000 images; 20 categories, 3,000 images; 20 categories, 6,000 images; 50 categories, 6,000 images; 50 categories, 15,000 images; 100 categories, 15,000 images; 100 categories, 30,000 images. The SUN Database currently contains 131,067 scene images and 908 scene categories. Due to the limitations of the experimental conditions, we selected 50,000 scene images as the experimental datasets from the SUN Database. These scene images are publicly available at http://groups.csail.mit.edu/vision/SUN/. The selected 50,000 images are constructed 5 datasets (using a combination of random selection by computers and artificial selection) and we name them sequentially as “Data 9” through “Data 13.” These datasets include the following number of scene images, respectively: 1,000 images, 5,000 images, 15,000 images, 30,000 images, and 50,000 images. In this way, a total of 80,000 scene images are selected as the experimental data in this paper and we have dealt with all experimental images into the format of $384 \times 256$ pixels.

5.2. Experimental Results and Analysis

5.2.1. Storage Performance Test and Analysis. When performing the storage performance test, we experimentally contrasted the time consumption of storing different scene image sets according to the node number of the Hadoop cluster. We conducted the performance test of the storage time when the number of nodes is 1, 2, 3, and 4. The experiment results are shown in Figure 4.

When the size of scene images is less than 5,000, the growth of node number does not clearly affect the performance of the amount of time consumed for scene image storage; when the size of scene images is more than 5,000, the performance on distributed parallel storage is more clearly observed. With the same size of the image database, the amount of time consumed for storage decreases with an increasing node number. When the size of the scene images becomes larger, the amount of time consumed for storage also becomes larger. However, the single node cluster increases most rapidly, and the rate of growth of the 4-node cluster is reduced. That is, when the image database is small, the use of a multinode cluster for distributed storage is not appropriate; when the image database is large, the efficiency of using distributed parallel storage is higher.

5.2.2. Retrieval Performance Test and Analysis. We developed an image retrieval prototype system based on Hadoop platform using Java. Figure 5 is the interface of the retrieval results aiming at the uploaded sample image retrieval.

To verify the retrieval performance, experiments are performed to compare the following three aspects: retrieval accuracy, retrieval time consuming, speedup and efficiency, sizeup, and scaleup.

(1) Retrieval Accuracy. Under different datasets of COREL Database and SUN Database, the traditional $k$-Means algorithm, $x$-Means algorithm in literature [24], global $k$-Means algorithm in literature [25], $k$-Means++ algorithm in literature [26], the parallel $k$-Means algorithm in literature [33],
and the proposed algorithm in this paper were compared in terms of their retrieval precision rates.

First, we made the experimental contrast using COREL Database and the experimental results are shown in Table 2.

The data in Table 2 show that, no matter which algorithm is used, the retrieval accuracy would decrease with the increase of image category under the circumstances of the same image number; and the retrieval accuracy would also decrease with the increase of image number under the circumstances of the same image category. This suggests that the larger the image number is and the more the image categories are, the more complicated the retrieval process would be, which leads to the decline of the retrieval accuracy. However, the growth of image categories has less influence on retrieval accuracy than the increase of image number, especially for the two parallel algorithms. In addition, it can be concluded that the retrieval accuracy would drop dramatically for the traditional algorithms (\(k\)-Means, \(x\)-Means, global \(k\)-Means, and \(k\)-Means++) with single node architecture, yet the retrieval accuracy of the two parallel algorithms declines slower, which fully reflects the superiority of distributed parallel computing. From the point of view of the statistical results of the standard deviation, the proposed algorithm in this paper has the fewest sample fluctuations and shows the best retrieval performance.

Next, we use the traditional quantities of precision rate, recall rate, and \(F_1\) value to evaluate the retrieval effect on datasets of SUN Database. The precision rate reflects the ability of rejecting irrelevant scene images according to the following formula:

\[
P_{\text{precision}} = \frac{\text{the number of retrieved relevant images}}{\text{the number of retrieved images}} \times 100%.
\]  

(4)

The recall rate reflects the ability of a system to classify relevant images according to the following formula:

\[
P_{\text{recall}} = \frac{\text{the number of retrieved relevant images}}{\text{the number of all relevant images in the retrieval system}} \times 100%.
\]  

(5)

For large-scale data analysis and retrieval, when the precision and recall rate appear contradictory, a comprehensive evaluation standard, namely, \(F\)-measure index is usually used to evaluate the system in order to evaluate retrieval performance. It combines the results of precision rate and recall rate and is the weighted harmonic mean of precision rate and recall rate. The higher the \(F\) value is, the better the retrieval performance is. The calculation formula is as follows:

\[
F = \frac{(\alpha^2 + 1) \times P_{\text{precision}} \times P_{\text{recall}}}{\alpha^2 \times (P_{\text{precision}} + P_{\text{recall}})},
\]  

(6)

where \(\alpha\) is adjusting parameter. When \(\alpha = 1\), that is the most commonly used \(F1\) value evaluation index:

\[
F1 = \frac{2 \times P_{\text{precision}} \times P_{\text{recall}}}{P_{\text{precision}} + P_{\text{recall}}},
\]  

(7)

It is generally believed that the higher \(F1\) value shows that the system reaches the optimal balance between precision rate and recall rate and achieves the better analysis and retrieval effectiveness.

Under different sizes of scene images, different algorithms appearing in Table 2 are compared in terms of their classification precision ratios, recall rates, and \(F1\) value. The experimental results are shown in Table 3 and Figure 6.

Table 3 shows the retrieval accuracy of different methods and Figure 6 shows the average retrieval accuracy of
Table 3: Contrast of the retrieval accuracy (%) of different methods based on SUN database.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Clustering algorithm</th>
<th>Precision rate (%)</th>
<th>Recall rate (%)</th>
<th>$F_1$ value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 9</td>
<td>$k$-Means</td>
<td>89.29</td>
<td>92.16</td>
<td>0.907</td>
</tr>
<tr>
<td></td>
<td>$x$-Means</td>
<td>91.47</td>
<td>94.06</td>
<td>0.927</td>
</tr>
<tr>
<td></td>
<td>Global $k$-Means</td>
<td>91.91</td>
<td>94.87</td>
<td>0.934</td>
</tr>
<tr>
<td></td>
<td>$k$-Means++</td>
<td>92.38</td>
<td>94.99</td>
<td>0.937</td>
</tr>
<tr>
<td></td>
<td>Parallel $k$-Means</td>
<td>92.54</td>
<td>95.43</td>
<td>0.940</td>
</tr>
<tr>
<td></td>
<td>The proposed method</td>
<td>96.48</td>
<td>98.87</td>
<td>0.977</td>
</tr>
<tr>
<td>Data 10</td>
<td>$k$-Means</td>
<td>86.41</td>
<td>89.25</td>
<td>0.878</td>
</tr>
<tr>
<td></td>
<td>$x$-Means</td>
<td>87.96</td>
<td>90.81</td>
<td>0.894</td>
</tr>
<tr>
<td></td>
<td>Global $k$-Means</td>
<td>87.83</td>
<td>90.81</td>
<td>0.893</td>
</tr>
<tr>
<td></td>
<td>$k$-Means++</td>
<td>88.37</td>
<td>91.04</td>
<td>0.897</td>
</tr>
<tr>
<td></td>
<td>Parallel $k$-Means</td>
<td>91.90</td>
<td>94.81</td>
<td>0.933</td>
</tr>
<tr>
<td></td>
<td>The proposed method</td>
<td>95.85</td>
<td>98.36</td>
<td>0.971</td>
</tr>
<tr>
<td>Data 11</td>
<td>$k$-Means</td>
<td>81.29</td>
<td>84.18</td>
<td>0.827</td>
</tr>
<tr>
<td></td>
<td>$x$-Means</td>
<td>82.35</td>
<td>85.02</td>
<td>0.837</td>
</tr>
<tr>
<td></td>
<td>Global $k$-Means</td>
<td>83.81</td>
<td>86.81</td>
<td>0.853</td>
</tr>
<tr>
<td></td>
<td>$k$-Means++</td>
<td>85.26</td>
<td>87.94</td>
<td>0.866</td>
</tr>
<tr>
<td></td>
<td>Parallel $k$-Means</td>
<td>90.86</td>
<td>93.83</td>
<td>0.923</td>
</tr>
<tr>
<td></td>
<td>The proposed method</td>
<td>94.93</td>
<td>97.89</td>
<td>0.964</td>
</tr>
<tr>
<td>Data 12</td>
<td>$k$-Means</td>
<td>74.10</td>
<td>76.92</td>
<td>0.755</td>
</tr>
<tr>
<td></td>
<td>$x$-Means</td>
<td>76.05</td>
<td>78.89</td>
<td>0.774</td>
</tr>
<tr>
<td></td>
<td>Global $k$-Means</td>
<td>76.72</td>
<td>79.28</td>
<td>0.780</td>
</tr>
<tr>
<td></td>
<td>$k$-Means++</td>
<td>79.31</td>
<td>81.69</td>
<td>0.805</td>
</tr>
<tr>
<td></td>
<td>Parallel $k$-Means</td>
<td>89.53</td>
<td>91.90</td>
<td>0.907</td>
</tr>
<tr>
<td></td>
<td>The proposed method</td>
<td>93.77</td>
<td>96.71</td>
<td>0.952</td>
</tr>
<tr>
<td>Data 13</td>
<td>$k$-Means</td>
<td>64.83</td>
<td>66.43</td>
<td>0.656</td>
</tr>
<tr>
<td></td>
<td>$x$-Means</td>
<td>66.59</td>
<td>67.91</td>
<td>0.672</td>
</tr>
<tr>
<td></td>
<td>Global $k$-Means</td>
<td>67.28</td>
<td>69.05</td>
<td>0.682</td>
</tr>
<tr>
<td></td>
<td>$k$-Means++</td>
<td>69.05</td>
<td>72.0</td>
<td>0.705</td>
</tr>
<tr>
<td></td>
<td>Parallel $k$-Means</td>
<td>88.14</td>
<td>91.03</td>
<td>0.896</td>
</tr>
<tr>
<td></td>
<td>The proposed method</td>
<td>92.49</td>
<td>95.59</td>
<td>0.940</td>
</tr>
</tbody>
</table>

Figure 6: Comparison of the average retrieval accuracies (%) based on SUN Database.

These phenomena describe that the improvement in the accuracy is not by chance. The optimization of initial clustering center of $k$-Means algorithm and the parallel design of the proposed method in MapReduce environment are important factors of improving the retrieval accuracy. In addition, $F_1$ value is greater than 0.9 and the average $F_1$ value is greater than 0.95 using the proposed method in this paper, which also describe that the proposed method reaches the very good balance between precision rate and recall rate and we have obtained desired retrieval performance.
In order to better observe the fluctuation of the retrieval accuracy, we drew a decline curve of the retrieval performance for different algorithms, as shown in Figures 7(a)–7(c).

Figures 7(a)–7(c) shows the decline curves of precision rate, recall rate, and $F_1$ value based on SUN Database. It can be concluded that the fluctuation of the proposed method in this paper and the parallel $k$-Means algorithm is much less than the traditional algorithms with single node architecture according to the curve change; moreover, as the data scale increases, although the retrieval accuracy rates of all methods decreased, the proposed method in this paper and the parallel $k$-Means algorithm obviously decreased only very slightly, which indicates that the parallel programming model based on MapReduce achieves an excellent performance level, particularly with large-scale data.

In addition, we also made the experimental contrast of the average retrieval accuracy for COREL Database (30,000 images), SUN Database (50,000 images), the mixture of COREL Database, and SUN Database (80,000 images) using different algorithms. The experimental results are shown in Figure 8.

Figure 8 is a sharp contrast of average retrieval accuracy of different methods in different datasets. When image number increases from 30,000 to 80,000, the average retrieval accuracies of the traditional algorithms ($k$-Means, $x$-Means, global $k$-Means, and $k$-Means++) with single node architecture all decrease by more than 25%, yet that of the parallel $k$-Means algorithm and the proposed method only decreases by 3.35% and 2.66%, respectively, which further shows that the retrieval performance of the traditional algorithms based on
Table 4: Comparison of retrieval times for the different methods based on COREL database.

<table>
<thead>
<tr>
<th>Retrieval method</th>
<th>Image category</th>
<th>3,000 images</th>
<th>6,000 images</th>
<th>15,000 images</th>
<th>30,000 images</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
<td>16</td>
<td>19</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>52</td>
<td>53</td>
<td>53</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,766</td>
<td>1,772</td>
<td>1,785</td>
<td>1,809</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6,258</td>
<td>6,297</td>
<td>6,322</td>
<td>6,413</td>
</tr>
<tr>
<td>k-Means</td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>51</td>
<td>53</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,752</td>
<td>1,770</td>
<td>1,784</td>
<td>1,801</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6,259</td>
<td>6,295</td>
<td>6,319</td>
<td>6,410</td>
</tr>
<tr>
<td>x-Means</td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14</td>
<td>14</td>
<td>17</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>51</td>
<td>54</td>
<td>56</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,750</td>
<td>1,776</td>
<td>1,788</td>
<td>1,800</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6,264</td>
<td>6,294</td>
<td>6,320</td>
<td>6,408</td>
</tr>
<tr>
<td>Global k-Means</td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15</td>
<td>15</td>
<td>18</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>47</td>
<td>54</td>
<td>56</td>
<td>59</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,758</td>
<td>1,769</td>
<td>1,783</td>
<td>1,803</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6,253</td>
<td>6,282</td>
<td>6,318</td>
<td>6,411</td>
</tr>
<tr>
<td>k-Means++</td>
<td></td>
<td>10</td>
<td>20</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15</td>
<td>15</td>
<td>18</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>47</td>
<td>49</td>
<td>53</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,758</td>
<td>1,769</td>
<td>1,783</td>
<td>1,803</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6,253</td>
<td>6,282</td>
<td>6,318</td>
<td>6,411</td>
</tr>
<tr>
<td>Parallel k-Means++ (using 4 slave nodes)</td>
<td>10</td>
<td>2.8</td>
<td>6.1</td>
<td>43</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>3.0</td>
<td>6.5</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>3.3</td>
<td>7.0</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
<td>3.9</td>
<td>7.8</td>
<td>50</td>
</tr>
<tr>
<td>The proposed method (using 4 slave nodes)</td>
<td>10</td>
<td>0.79</td>
<td>1.3</td>
<td>10</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>0.80</td>
<td>1.5</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
<td>0.83</td>
<td>2.0</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
<td>0.84</td>
<td>2.7</td>
<td>15</td>
</tr>
</tbody>
</table>

The proposed method (using 4 slave nodes)

Figure 8: Average retrieval accuracy (%) comparison of different methods.

Figure 9: Average retrieval rate (%) comparison of different methods.

(2) Retrieval Time Consuming. To further verify the effectiveness of the proposed method in this paper, we make the experimental contrast of retrieval time. The experimental results are shown in Tables 4 and 5.

Furthermore, to test the time performance of the method proposed in this paper in MapReduce environment, we randomly selected 60,000 images from 80,000 images database (the mixture of COREL Database and SUN Database) to make comparison in the case of different slave node computers only using the parallel $k$-Means algorithm and the proposed method in this paper. The experimental results are shown in Figure 9.

Tables 4 and 5 present the contrast results of retrieval time consuming for different data scale using different methods. Aiming at the parallel computing environment, Figure 9 compares the time performances of the two parallel
Table 5: Comparison of retrieval times for the different methods based on SUN database.

<table>
<thead>
<tr>
<th>Image number</th>
<th>$k$-Means</th>
<th>$x$-Means</th>
<th>Global $k$-Means</th>
<th>$k$-Means++</th>
<th>Parallel $k$-Means</th>
<th>The proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>0.9</td>
<td>0.8</td>
<td>0.9</td>
<td>0.9</td>
<td>0.2</td>
<td>0.026</td>
</tr>
<tr>
<td>5,000</td>
<td>51</td>
<td>53</td>
<td>54</td>
<td>54</td>
<td>7</td>
<td>1.2</td>
</tr>
<tr>
<td>15,000</td>
<td>1,793</td>
<td>1,796</td>
<td>1,794</td>
<td>1,799</td>
<td>54</td>
<td>13</td>
</tr>
<tr>
<td>30,000</td>
<td>6,257</td>
<td>6,285</td>
<td>6,277</td>
<td>6,289</td>
<td>110</td>
<td>69</td>
</tr>
<tr>
<td>50,000</td>
<td>12,109</td>
<td>12,132</td>
<td>12,133</td>
<td>12,129</td>
<td>187</td>
<td>149</td>
</tr>
</tbody>
</table>

Explanation: The parallel $k$-Means algorithm and the proposed method in this paper used 4-slave-node computers.

Figure 9: Comparison of retrieval time consuming for randomly selected 60,000 images.

algorithms under different computing nodes. We note that the retrieval time of the traditional algorithms ($k$-Means, $x$-Means, global $k$-Means, and $k$-Means++) is much longer than that of the other two parallel algorithms. This is because the two parallel algorithms adopt distributed parallel processing technology, while the traditional algorithms use the single node architecture with low processing capacity. As can be seen in Tables 4 and 5 and Figure 9, the retrieval time of the method proposed in this paper is less than that of the parallel $k$-Means algorithm, which is mainly because the method proposed in this paper not only optimizes the initial clustering center of $k$-Means algorithm, but also optimizes the iterative process of the algorithm in the MapReduce environment. Therefore, desired retrieval time performance is obtained.

Figure 8 shows the contrast results of retrieval time consuming for 50,000 scene images in different nodes using parallel $k$-Means algorithm and the proposed method, respectively.

(3) Speedup and Efficiency, Sizeup, and Scaleup. For the parallel programming model based on MapReduce, we evaluate the computational performance of the proposed algorithm in terms of speedup and efficiency, sizeup, and scaleup. To test the computational performance, we randomly selected 5 different datasets: 5,000 images, 10,000 images, 20,000 images, 50,000 images, and 80,000 images.

Speedup refers to the ratio of the time required to run a task on a single calculating node to the time required to run that same task on multiple calculating nodes, while efficiency refers to the ratio of the speedup to the number of calculating nodes [14]. In an ideal state, the speedup should increase linearly and the efficiency should remain constant with the increase of the number of nodes. However, the efficiency does not reach 1 because task control is influenced by communication cost and load balance, among other factors. Goller et al. [44] suggested that the system has obtained good performance as long as the efficiency can reach 0.5. Figures 10(a) and 10(b) show experimental comparisons of the speedup ratios and of the efficiencies of the method proposed in this paper, respectively, using datasets of different scales.

In Figure 10(a), the speedup ratio takes on a rising tendency as the number of calculating nodes increases, and increased data scale results in an increased magnitude of the speedup ratio. For the same dataset, the processing speed of the system becomes faster with the increase of the number of computing nodes; that is to say, the processing time become less, so the speedup ratio takes on a rising tendency. For different datasets, the larger the amount of data is, the better the performance of the multicomputing nodes is. So compared to the single node computer, the processing speed would be much faster. This result further indicates that larger datasets better demonstrate the performance of multiple calculating nodes. Figure 10(b) validates the retrieval system’s efficiency.

As the number of calculating nodes increases, the system efficiency decreases more rapidly than when the dataset is smaller. As the size of the dataset gradually increases, the increase in calculating nodes results in a reduced system efficiency, yet with a smaller magnitude. The main reason for the reduction in system efficiency is that the increase in data size causes the system processing time to increase. In contrast, as the number of calculating nodes increases, the communication overhead between nodes also increases. However, the system efficiency is always above 0.5, which indicates that the method has excellent parallel performance and expandability.

Sizeup is defined as how much longer it takes on a given system when the image size is $m$-times larger than the original
image size. That is to say, with the increase of image number, the higher the value of sizeup is, the longer the time the cluster system would take. To measure sizeup, we fix the number of the slave nodes to 1, 2, 3, and 4, respectively, and increase the image size from 5,000 images to 80,000 images for each node number. Figure 11 shows the experimental results. It is easy to see that when we increase image size from 5,000 to 80,000, the sizeup of 1-node system is increased by about 3 while it is only increased by 2.3 for 4-node system. This is because the communication time of the 1-node system is smaller than that of the 4-node system, and the communication time would not increase much for the proposed method in this paper when the image number is increased. Therefore, the proposed method in this paper has a good sizeup performance.

Scaleup refers to the ability of an $m$-times larger system to perform an $m$-times larger job in the same run time as the original system, which is used to evaluate the ability of the algorithm to grow both the system and the image size. Obviously, the higher value of the scaleup indicates the better algorithm performance. Therefore, scaleup validates how well the algorithm handles larger image size when more node computers are available. For the measure of scaleup, we increase the node computers and image size simultaneously and obtain the different scaleup values for the following combinations (1-node computer, 5,000 images), (2-node computers, 10,000 images), (3-node computers, 50,000 images), and (4-node computers, 80,000 images). The results are shown in Figure 12. The higher the scaleup value, the better the performance we would obtain. The results in
Figure 12 show that the values of scaleup are all higher than 0.90, which demonstrates the proposed method scales well.

6. Conclusions

As a kind of media with visual and integrated information, digital images have more and more penetrated and served in all fields and gradually influence people's work, study, and life. Accordingly, it has become the urgent problem to be solved to organize, manage, and analyze large-scale image data. Scene images are very common image data, which contains extremely rich content. Therefore, how to make the computer understand the image contents like the human and make retrieval results meet users' needs is very important. At the same time, building parallel retrieval framework for large-scale scene images based on big data technology to improve the retrieval efficiency would lay a solid foundation for the effective organization and management of massive image data.

This paper discussed the development of a large-scale scene image retrieval method based on an improved parallel \( k \)-Means feature cluster algorithm and better improved the time efficiency and retrieval accuracy for large-scale image retrieval. It studied how to improve the parallel \( k \)-Means algorithm and apply it to feature cluster retrieval for scene images. Finally, retrieval for large-scale scene images was realized using the Hadoop distributed processing platform. The experimental results demonstrated that the designed scheme could balance the system load, make full use of the resources of the distributed system, and improve the retrieval speed. When considering large-scale scene images, the retrieval efficiency of the Hadoop distributed system was greatly improved relative to the retrieval efficiency of a system of single-node architecture, which fully embodied the powerful computing capability of the distributed parallel processing architecture.

With the development of parallel technology, parallel computing plays more and more important role in dealing with the complex problems of huge amount calculations. The purpose of this paper is to apply MapReduce parallel programming framework to traditional \( k \)-Means algorithm optimized by Canopy algorithm so as to improve the clustering speed of \( k \)-Means algorithm through the research on the parallel processing technology of Hadoop cluster. In the field of digital image understanding, using the powerful data processing ability of parallel computing to mine and analyze massive data is helpful to obtain more accurate image information. It has very important value for image annotation, classification, and retrieval. It is of great significance to improve the intelligence for digital image understanding.

Analysis and processing of large-scale multimedia data have become popular research topics with the arrival of the big data era and the development of cloud computing and multimedia technology. Future research to improve the results of this paper mainly includes the following: (1) expansion of the node number of the Hadoop distributed platform, adjustment of the relevant parameters of the system, and further improvements to the efficiency of the distributed system; (2) optimization of the feature extraction and clustering algorithm to improve the retrieval accuracy; and (3) optimization of the task design of Map and Reduce to achieve faster and more precise retrieval.

Competing Interests

The authors declare no conflict of interests.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (61202163), the Shanxi Natural Science Foundation (2013011017-2 and 2014011019-3), the Science and Technology Innovation Project of Colleges and Universities in Shanxi Province (2013350), and the Specialized Subject for Key Disciplines of Xinzhou Teachers University, China (XK201308, XK201404).

References


16 Mathematical Problems in Engineering


