Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2014, Article ID 430814, 8 pages
http://dx.doi.org/10.1155/2014/430814

Research Article

A Distributed Weighted Possibilistic c-Means Algorithm for Clustering Incomplete Big Sensor Data

Qingchen Zhang and Zhikui Chen

School of Software Technology, Dalian University of Technology, Liaoning, China

Correspondence should be addressed to Qingchen Zhang; 623759909@qq.com

Received 4 March 2014; Revised 26 April 2014; Accepted 27 April 2014; Published 19 May 2014

Academic Editor: Jianwei Niu

Possibilistic c-means clustering algorithm (PCM) has emerged as an important technique for pattern recognition and data analysis. Owning to the existence of many missing values, PCM is difficult to produce a good clustering result in real time. The paper proposes a distributed weighted possibillistic c-means clustering algorithm (DWPCM), which works in three steps. First the paper applies the partial distance strategy to PCM (PDPCM) for calculating the distance between any two objects in the incomplete data set. Further, a weighted PDPCM algorithm (WPCM) is designed to reduce the corruption of missing values by assigning low weight values to incomplete data objects. Finally, to improve the cluster speed of WPCM, the cloud computing technology is used to optimize the WPCM algorithm by designing the distributed weighted possibilistic c-means clustering algorithm (DWPCM) based on MapReduce. The experimental results demonstrate that the proposed algorithms can produce an appropriate partition efficiently for incomplete big sensor data.

1. Introduction

Recent years have witnessed the deployments of sensor networks for many critical applications such as Internet of Things (IoT) [1, 2], environment monitoring [3], and target detection [4, 5]. With the rapid advent of various sensor networks, more and more mobile devices, such as smart phones and RFID, can sense and collect a huge number of sensor data. For example, data sampled from millions of sensors deployed sensor networks for environment monitoring can exceed hundreds of terabytes data every day. We are moving toward the era of big sensor data which requires efficiently advanced data analysis and mining tools [6].

As an important data mining tool, the possibilistic c-means cluster algorithm (PCM) has emerged as an important technique for pattern recognition and data analysis, proposed by Krishnapuram and Keller [7]. Many PCM variants have been proposed after standard PCM for improving the performance of the original PCM algorithm. Zhang and Leung applied the fuzzy method to PCM for an improved PCM algorithm, which enhances the robustness of the possibilistic approach [8]. In 2011, Yang and Lai proposed a robust merging approach and created the automatic merging possibilistic clustering method to obtain the number of classes automatically. However, this algorithm could not always obtain the most reasonable number of clusters [9]. Pal proposed FPCM and PFCM, which is a combined way of doing PCM with FCM, avoiding the coincidental cluster [10, 11]. In 2008, Xie et al. proposed an enhanced possibilistic c-means clustering algorithm, which partitioned the dataset into the main cluster and the assisted cluster to avoid producing the coincidental cluster [12].

PCM can be extensively used in big sensor data analysis and mining. However, in big sensor data, many data sets suffer from incompleteness; that is, a data set X can contain vectors that miss one or more of the attribute values [13]. PCM could not succeed completely in clustering such incomplete data sets in real time. On the one hand, PCM could not calculate the distance between two objects in incomplete data sets. Meanwhile the accuracy of PCM is easily corrupted by incomplete objects. On the other hand, PCM is difficult to satisfy the real-time requirement of clustering

incomplete big sensor data due to the huge amount of data.

The paper proposes a distributed weighted possibilistic c-means algorithm (DWPCM) for clustering incomplete big sensor data. First, the paper applies the partial distance strategy (PDS) [14] to PCM (PDPCM) for calculating the distance between two objects in incomplete data set using all available attribute values. Further, a weighted PDPCM algorithm (WPCM) is designed to reduce the corruption of missing values. In WPCM, a novel method for determining weight values is presented. Finally, the cloud computing technology is used to optimize the WPCM algorithm by designing the distributed weighted possibilistic c-means algorithm (DWPCM) based on MapReduce, which aims at improving the cluster speed of WPCM.

To evaluate the performance of the proposed algorithms, we implement the proposed methods on two real big sensor data sets. The experimental results demonstrate that the proposed algorithms can produce an appropriate partition efficiently for incomplete big sensor data.

The rest of this paper is structured as follows. Section 2 resumes some related research on possibilistic c-means clustering algorithms and the cluster algorithms based on cloud computing. Section 3 depicts the WPCM algorithm and Section 4 describes the DWPCM algorithm based on MapReduce. The performance evaluation is illustrated in Section 5. The paper is concluded in Section 6.

2. Related Work

2.1. Possibilistic c-Means Algorithms. PCM partitions an m-dimensional dataset $X = \{x_1, x_2, \ldots, x_n\}$ into several clusters to describe an underlying structure within the data. A possibilistic partition is defined as a $c \times n$ matrix $U = \{u_{ij}\}$, where u_{ij} is the membership value of object x_j towards the ith cluster, c is the number of clusters, and n is the number of objects. PCM obeys the constraint as follows:

$$u_{ij} \in [0,1] \quad \forall i, j, \quad 0 < \sum_{j=1}^{n} u_{ij} \leq N \quad \forall i \,, \qquad \max_i u_{ij} > 0 \quad \forall j. \tag{1}$$

PCM minimizes the following objective function [7]:

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m \|x_k - v_i\|^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} \left(1 - u_{ij}\right)^m. \tag{2}$$

In (2), $V = (v_1, v_2, \ldots, v_c)$ is a C-tuple of prototypes and $U = [u_{ij}]$ is a $c * n$ matrix, called the possibilistic c-partition matrix, satisfying the conditions in (1). Here, $m > 1$ is a fuzzification constant and η_i is a suitable positive number. The first term demands that the distances from the object to the prototypes be as low as possible, whereas the second term forces the u_{ij} to be as large as possible, aiming at avoiding the trivial solution.

Solving the minimization problem (2) yields membership functions of the form

$$u_{ij} = \frac{1}{1 + \left(d_{ij}/\eta_i\right)^{1/(m-1)}}. \tag{3}$$

The cluster centers are updated using

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}. \tag{4}$$

The procedure of PCM can be described as follows.

Step 1. Choose m, c, and $\varepsilon > 0$ and then initialize the membership matrix $U^{(0)}$.

Step 2. Update cluster centers using (4).

Step 3. Estimate η_i using the following formula:

$$\eta_i = \frac{\sum_{j=1}^{n} u_{ij}^m d_{ij}^2}{\sum_{j=1}^{n} u_{ij}^m}. \tag{5}$$

Step 4. Update membership matrix U using (3).

Step 5. If $\varepsilon \leq \|u_{ij} - u'_{ij}\|^2$, stop; else repeat Step 2.

PCM is rational and it has a meaningful interpretation. What is more important is that PCM is robust, because a noisy object would belong to the clusters with small memberships, and consequently it cannot corrupt the resulting clusters significantly [8].

In the PCM algorithm, each object is considered equally important in the clustering solution. The weighted PCM (wPCM) model introduces weights to define the relative importance of each object in the clustering solution [15, 16]. Assume that $\mathbf{w} \in R^n$, $w_j \geq 0$, is a set of weight values that define the influence of each object, which results in the WPCM objective function of the form

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m \|x_k - v_i\|^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} \left(w_j - u_{ij}\right)^m. \tag{6}$$

In order to find clusters with nonspherical shapes, kernel-based PCM algorithms are proposed, which mapped original data into higher dimensional feature space [17]. Given a kernel function k, the kernel PCM algorithm (kPCM) minimizes the following objective function:

$$J_m(U,V;k) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m \Phi_{ij}^2 + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{n} \left(1 - u_{ij}\right)^m, \tag{7}$$

where Φ_{ij} is the kernel-based distance between the ith object and the jth object.

Even though these methods are performed well in the cluster process, they cannot succeed completely in clustering incomplete big sensor data because of the corruption of missing values. The paper proposed a weighted PCM algorithm (WPCM), which applies the partial distance to PCM for calculating the distance between two objects in incomplete data set and then assigns low weight values to incomplete data object for reducing the corruption of missing values.

2.2. Cluster Algorithms Based on Cloud Computing. Cloud computing has emerged as a significant technology to deal with big data in time by leveraging vast amounts of computing resources available on demand with low resource usage cost [18, 19]. As the key technology of cloud computing, MapReduce [20] is a highly efficient distributed programming model for large-scale data sets in parallel computing.

Many cluster algorithms based on MapReduce have been proposed to improve the cluster efficiency. For example, Zhao [21] proposed parallel k-means algorithm based on MapReduce to cluster massive data fast. Similar algorithms include MapReduce-kCenter algorithm and MapReduce-kMedian algorithm, which used iterative sampling technology to get good performance for clustering very large data [22, 23]. Another example is hierarchical clustering algorithm based on cloud computing, in which MapReduce is used to optimize the hierarchical clustering algorithm for processing large-scale data [24, 25]. Well-known cluster algorithms based on MapReduce, such as HDBSCAN [26] and MR-DBSCAN [27], reduce I/O access frequency and spatial complexity for speeding up the density clustering. For affinity propagation clustering algorithm, published in science magazine proposed by Frey and Dueck [28], Lu et al. [29] proposed a distributed AP clustering algorithm based on MapReduce (DisAP), which includes three MapReduce stages and achieved high performance on both scalability and accuracy. In 2012, Yang et al. [30] presented a MapReduce-based MST text clustering algorithm, which used cloud computing technology to improve the performance of the graph clustering. Yu and Dai [31] proposed a parallel fuzzy c-means algorithm based on MapReduce for improving the standard fuzzy c-means algorithm. Other parallel clustering algorithms based on cloud computing can be found in [32–34].

Even though these algorithms perform their job well, they all focus on crisp clustering. The paper uses cloud computing technology to accelerate the cluster speed of WPCM by designing a distributed WPCM algorithm (DWPCM) based on MapReduce to produce the possibilistic clustering for big data in real time.

3. Weighted PCM Algorithm

3.1. PCM Based on Partial Distance for Clustering Incomplete Data. In this subsection, the paper applies partial distance to PCM for clustering incomplete data set. Partial distance is used to calculate the distance between an object x_k and the ith cluster center v_i as follows:

$$\mathrm{PD}_{ik} = \frac{m}{I_k}\sqrt{\sum_{j=1}^{m}\left(x_{kj} - v_{ij}\right)^2 I_{kj}} \tag{8a}$$

$$I_{kj} = \begin{cases} 0, & \text{if } x_{kj} = * \\ 1, & \text{otherwise} \end{cases}$$

$$\text{for } 1 \le j \le m,\ 1 \le k \le n; \tag{8b}$$

$$I_k = \sum_{j=1}^{m} I_{kj}.$$

From (8a) and (8b), partial distance makes full use of attribute information of both complete data and incomplete data to calculate the distance between two objects.

The PCM algorithm based on partial distance (PDPCM) is obtained by making two modifications of PCM: (1) calculating d_{ij} for incomplete data according to (8a) and (8b) and (2) updating the cluster centers with

$$v_{ij} = \frac{\left(\sum_{k=1}^{n} u_{ik}^{m} I_{kj} x_{kj}\right)}{\left(\sum_{k=1}^{n} u_{ik}^{m} x_{kj}\right)}. \tag{9}$$

This algorithm enjoys all the standard convergence properties of PCM because it is an instance of alternating optimization [7].

For an m-dimensional incomplete data set, the procedure of the PCM algorithm for clustering incomplete data based on partial distance can be described as follows.

Step 1. Choose m, c, and $\varepsilon > 0$ and then initialize the membership matrix $U^{(0)}$.

Step 2. Update cluster centers using (9).

Step 3. Estimate η_i using (5).

Step 4. Update membership matrix U using (3).

Step 5. If $\varepsilon \le \|u_{ij} - u'_{ij}\|^2$, stop; else repeat Step 2.

3.2. Weighted PCM Algorithm. Even though the PCM algorithm based on partial distance given earlier can cluster incomplete data sets, it is difficult to produce a good partition due to the effect of incomplete objects. The paper proposes a weighted PCM algorithm (WPCM), which assigns low weight values to incomplete objects for reducing the corruption of incomplete objects on the cluster process.

To minimize the objective function, WPCM updates the membership values and clusters centers using the following equation:

$$\eta_i = \frac{\sum_{j=1}^{n} w_j u_{ij}^m d_{ij}^2}{\sum_{j=1}^{n} w_j u_{ij}^m}, \tag{10}$$

$$u_{ij} = \frac{w_j}{1 + \left(d_{ij}/\eta_i\right)^{1/(m-1)}}, \tag{11}$$

$$v_i = \frac{\sum_{j=1}^{n} w_j u_{ij}^m x_j}{\sum_{j=1}^{n} w_j u_{ij}^m}. \tag{12}$$

The w_j weights can be viewed as strength terms describing the strength of membership of x_j in any cluster [16]. Recently, a large number of algorithms have been proposed to determine the weight of the object. Perhaps the most representative method for detecting the outlier and reducing their effect on the cluster process in PCM was proposed by Schneider [16], which determines the weight of each object depending on the degree of belonging of each feature vector in any cluster. The equation for the weight values is

$$s_j = \sum_{k=1}^{c} e^{-\alpha \|x_j - v_k\|^2}, \quad j = 1, 2, \ldots, n, \tag{13}$$

where $a > 0$ is a suitably chosen constant.

Equation (13) can detect the outlier effectively. However, it cannot reduce the effect of incomplete objects on the cluster process. The paper redefines the weights in

$$w_j = \left(1 - \frac{l_j}{m}\right)^t \sum_{k=1}^{c} e^{-\alpha \|x_j - v_k\|^2}, \quad j = 1, 2, \ldots, n, \tag{14}$$

where m is the number of the features of the data object, t represents the iterative times, and l_j is the number of missing feature values of the data object x_j. Equation (14) reduces the corruption of incomplete objects on the cluster process by the coefficient $(1 - l_j/m)$. A large l_j value of x_j, which indicates that x_j has many missing feature values, will result in a low weight value. In (14), t increases with the times of the cluster iteration increasing, which can accelerate the convergence of the clustering process.

The steps of the WPCM algorithm are outlined as follows.

Step 1. Choose m, c,and $\varepsilon > 0$ and then initialize the cluster centers $V^{(0)}$ and the membership matrix $U^{(0)}$.

Step 2. Calculate the weight values using (14).

Step 3. Estimate η_i using (10).

Step 4. Update membership matrix U using (11).

Step 5. Update the cluster centers V using (12).

Step 6. If $\varepsilon \le \|u_{ij} - u'_{ij}\|^2$, stop; else repeat Step 2.

3.3. Time Complexity. In this subsection, we discuss the time complexity of WPCM. All operations are counted as unit costs. We do not assume time economies that might be realized by special programming tricks or properties of the equations involved. We use the following notation in our discussion:

i is the number of iterations of WPCM over the full data set;

n is the number of data objects;

f is the number of dimensions;

c is the number of clusters.

From the cluster process of the WPCM algorithm, the time complexity of this algorithm is dominated by Steps 4 and 5. The calculation of membership matrix using (14) requires $O(ncf)$ operations per cluster, which is also required by updating the cluster centers, resulting in a total time complexity of $O(incf)$ for WPCM.

4. Distributed Weighted PCM Algorithm Based on MapReduce

In Section 3, we propose a weighted PCM algorithm based on partial distance, which can cluster incomplete data effectively. However, WPCM has a low efficiency for clustering incomplete big sensor data owning to the huge number of data and its high time complexity.

To accelerate the cluster speed of WPCM, the paper proposes a distributed WPCM algorithm (DWPCM) based on MapReduce in this section.

From the steps of WPCM, there are two major operations: calculating the degree of membership u_{ij} and calculating the clustering centers v_i.

In the map phase, the Map function is designed to calculate the degree of membership u_{ij}.

To reduce the communication cost of the distributed algorithm, the paper partitions the membership matrix into p blocks by columns, each block with n/p columns, where n is the number of the data objects and p is the number of the data nodes in a cloud platform. After partition, the paper puts each block of the membership matrix in a data node to calculate.

In order to update cluster centers in parallel, two parameters, $\xi_i^{(t)}$ and $\lambda_i^{(t)}$, are introduced, where t represents the serial number of data node. After calculating the membership u_{ij}, the Map function calculates $\xi_i^{(t)}$ and $\lambda_i^{(t)}$ using

$$\xi_i^{(t)} = \sum_{k=1}^{n/p} w_k u_{ik}^m x_k \quad i = 1, 2, \ldots, c \tag{15}$$

$$\lambda_i^{(t)} = \sum_{k=1}^{n/p} w_k u_{ik}^m \quad i = 1, 2, \ldots, c. \tag{16}$$

Finally, the Map function outputs c $\langle key', value' \rangle$, where c is the number of classes, key' represents the identifier of the class, and $value'$ is a vector that consists of $\xi_{key'}^{(t)}$ and $\lambda_{key'}^{(t)}$.

Table 1: Averaged results of 10 trials on the eGASD set in terms of E_*.

Missing ratio	PDPCM	WPCM	DWPCM
1%	36.24	13.09	13.09
3%	42.98	17.26	17.26
5%	48.85	20.12	20.12
10%	51.07	24.78	24.78
15%	59.69	30.24	30.24
20%	66.94	32.58	32.58

Table 2: Averaged results of 10 trials on the sWSN set in terms of E_*.

Missing ratio	PDPCM	WPCM	DWPCM
1%	1.19	0.54	0.54
3%	1.52	0.69	0.69
5%	1.82	0.92	0.92
10%	2.07	1.15	1.15
15%	2.48	1.27	1.27
20%	2.97	1.54	1.54

In the reduce phase, the Reduce function is designed to calculate the clustering centers v_i.

The input of the Reduce function is a $\langle key', list\rangle$, where key' is the identifier of the class and $list$ includes all of *value's* with the same key' obtained from the Map function. The Reduce function is responsible to compute the cluster centers according to

$$v_i = \frac{\sum_{t=1}^{p} \xi_i^{(t)}}{\sum_{t=1}^{p} \lambda_i^{(t)}}, \tag{17}$$

where p is the number of the data nodes and i is the identifier of the class, which have the same explanation with key'.

Similar to Section 3.3, we discuss the time complexity of DWPCM now. The notation used in our discussion is the same as that in Section 3.3. The time complexity of DWPCM is approximately $O(incf/p)$, where p is the number of data nodes in the cloud computing platform. Note that DWPCM sends the clusters and the weight values to all the data nodes in each iteration, which may cause communication overhead. However, the communication takes significantly less time than the calculation on the cluster process, especially in a centralized cloud computing platform where all cloud computing nodes are in the same location. So we ignore the communication overhead when estimating time complexity.

5. Experiments

In order to evaluate the efficiency and effectiveness of the proposed algorithms, we perform the algorithms on the two real data sets. The experimental setup and dataset are described first, followed by the results.

5.1. Experimental Setup and Dataset. The experimental environment consists of 20 computers as cloud computing nodes, each of which has an Intel Core i7 processor with 3.2 GHz speed, 8 GB RAM, and a 2 TB hard drive.

We apply the algorithms on two real data sets in the experiments. The first data set is an extension of "Gas Sensor Array Drift Dataset at Different Concentrations Data Set" which is available in UCI Machine Learning Repository [35, 36]. This data set contains 3×10^9 objects with 129 numerical attributes, called eGSAD. The second data set consists of 2×10^{10} objects sampled from the smart WSN lab, called sWSN, whose size is about 1 TB. The parameters used for the proposed algorithms are $\varepsilon = 10^{-3}$ and $m = 1.75$, which can help to produce a good cluster result [37].

Table 3: Averaged results of 10 trials on eGASD in terms of ARI.

Missing ratio	PDPCM	WPCM	DWPCM
1%	0.9561	0.9893	0.9893
3%	0.9347	0.9754	0.9754
5%	0.9021	0.9556	0.9556
10%	0.8684	0.9331	0.9331
15%	0.8249	0.9092	0.9092
20%	0.7627	0.8861	0.8861

Table 4: Averaged results of 10 trials on sWSN in terms of ARI.

Missing ratio	PDPCM	WPCM	DWPCM
1%	0.9236	0.9653	0.9653
3%	0.8742	0.9238	0.9238
5%	0.8426	0.8958	0.8958
10%	0.8182	0.8630	0.8630
15%	0.7587	0.8204	0.8204
20%	0.7121	0.7862	0.7862

We first artificially create missing values in the data sets for simulating incomplete data sets and then cluster them using the proposed algorithms. The performance of the proposed algorithms is evaluated by comparing their cluster results to the result of PCM for clustering original data sets from both the efficiency and effectiveness points of view. While the efficiency concerns the time required to cluster the data sets, the effectiveness is related to the cluster accuracy.

Since the cluster performance depends on the amount of missing values, we artificially create six kinds of missing ratios, which are 1%, 3%, 5%, 10%, 15%, and 20% objects with missing values. For every missing ratio, we generate 5 different incomplete data sets for Gisette and sWSN. Specifically, any two data sets can have different missing values for every missing ratio.

5.2. Experimental Results on Cluster Accuracy. In order to assess the effectiveness of WPCM, two well-known evaluation criteria, E_* and Adjusted Rand Index (ARI), are used in the experiment [37, 38].

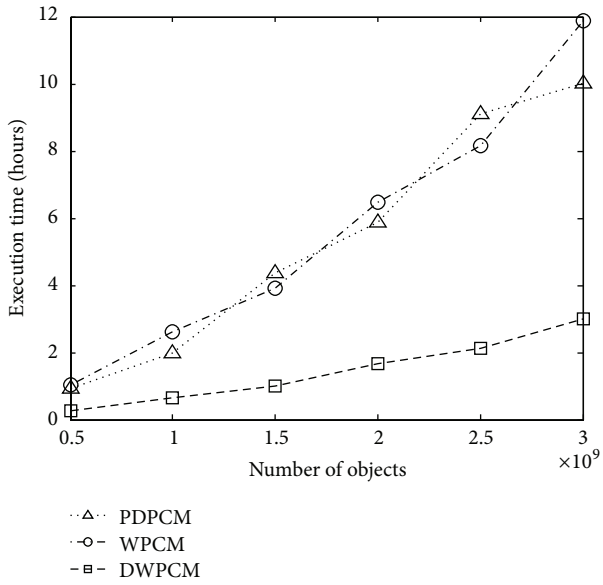

Figure 1: Average execution time on eGASD.

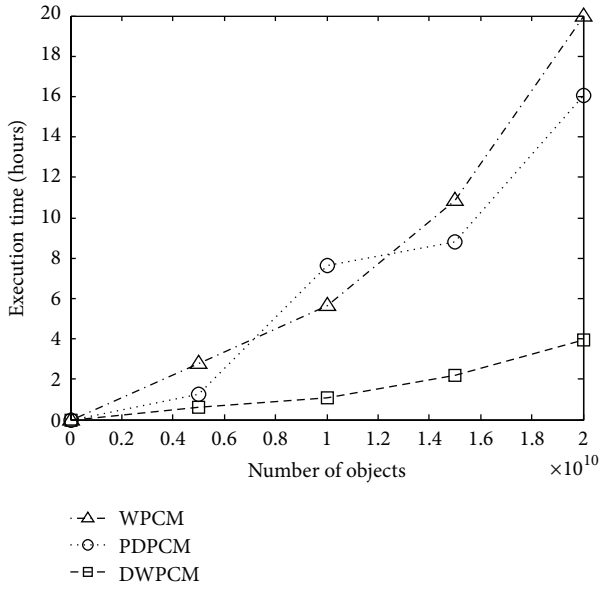

Figure 2: Average execution time on sWSN.

The first evaluation criterion, E_*, is used to assess the error between ideal cluster centers and cluster centers produced by a specific algorithm, which is calculated according to

$$E_* = \sqrt{\sum_{i=1}^{c} \left\| v_{\text{ideal}}^i - v_*^i \right\|^2}, \tag{18}$$

where v_{ideal}^i represents the ith ideal cluster center and v_*^i denotes the ith cluster center produced by a specific algorithm $*$. A lower value E_* of indicates that the algorithm produces more accurate cluster centers.

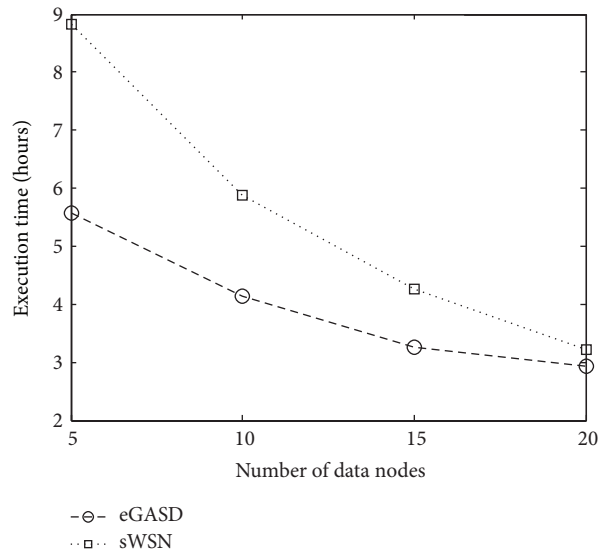

Figure 3: Computation speed of DWPCM.

The other evaluation criterion, ARI(U, U'), is used to measure the agreement between two possibilistic partitions of a set of objects, where U represents the ground truth labels for the objects in the data set and U' denotes a partition produced by a specific algorithm. A higher value of ARI(U, U') indicates that the algorithm produces a better cluster result.

To eliminate the variation in the results from trial to trial, Tables 1 and 2 present the average values of E_* and ARI obtained over 10 trials on incomplete eGASD and sWSN data sets, and the same incomplete data set is used in each trail for each of the three approaches so that the results can be correctly compared.

We present the cluster accuracy of PCM, PCM based on partial distance (PDPCM), WPCM, and DWPCM on the two data sets in terms of E_* for 6 missing ratios in Tables 1 and 2.

From Tables 1 and 2, as the missing ratio increases, the average values of E_* of three algorithms increase, which argues that the cluster accuracy is affected by missing ratios. In terms of E_*, WPCM always performs better than PDPCM because the average E_* value of WPCM is lower than that of PDPCM for 6 missing ratios, which demonstrates that the cluster prototypes obtained by WPCM are closer to the actual ones. DWPCM produces the same result as WPCM based on E_* because the two algorithms use the same methods to calculate the weight values, the cluster prototypes, and the membership matrix.

To calculate the ARI, we first harden the possibilistic partitions by setting the maximum element in each column of U to 1 and all else to 0. Tables 3 and 4 show the average values of ARI(U, U') obtained by PDPCM, WPCM, and DWPCM.

From the results shown in Tables 3 and 4, WPCM produces better partitions than PDPCM for 6 different missing ratios of the two data sets in terms of ARI. DWPCM still produces the same result as WPCM based on ARI.

5.3. Experimental Results on Execution Time. We use execution time and scalability to evaluate the efficiency of DWPCM. The average execution time of PDPCM, WPCM, and DWPCM on the two data sets for different number of objects is shown in Figures 1 and 2.

From Figures 1 and 2, even though the average execution time of the three algorithms increases with the number of objects increasing, DWPCM takes the least time of the three algorithms for the two data sets. Especially when the data set is very big, the execution time required by DKPCM is significantly less than the other two algorithms, which demonstrates that DWPCM performs most efficiently for clustering big sensor data.

In order to test the scalability of DWPCM, we perform the algorithm for clustering the two data sets in the different cloud computing platforms, in which there are 5 nodes, 10 nodes, 15 nodes, and 20 nodes, respectively. The result is shown in Figure 3.

From Figure 3, the execution time of DWPCM for clustering the two data sets reduces gradually with the increasing number of data nodes in the cloud computing platform, which demonstrates that adding nodes can significantly improve the system capacity. Therefore, DWPCM has a good scalability, especially when the data set is big.

6. Conclusion and Future Work

The paper proposes a distributed weighted PCM algorithm for clustering incomplete big sensor data. The proposed algorithm applies partial distance strategy to PCM (PDPCM) for calculating the distance between any two objects in the incomplete data set. Further, based on PDPCM, the paper designs a weighted PCM algorithm (WPCM) to reduce the corruption of incomplete objects on the cluster process. Another unique property of the proposed algorithm is the use of the cloud computing technology. Cloud computing is used to optimize WPCM to provide a significant computation speed, which is very important for big sensor data real-time clustering and analysis.

The experiments demonstrate that WPCM produces a good cluster result based on both evaluation criteria, namely, E_* and ARI. As for efficiency, DWPCM performs better than WPCM, which takes significantly less time than WPCM for clustering incomplete big sensor data. Note that DWPCM produces the same cluster result as WPCM in terms of E_* and ARI, which demonstrates that DWPCM improves the cluster efficiency without reducing the cluster accuracy.

In the future research work, we will investigate a further improvement of DWPCM to improve the effectiveness and efficiency of clustering big sensor data with many missing values. Additionally, for many semistructured and unstructured data in big sensor data, our future research plans will modify DWPCM to cluster the two types of incomplete data into appropriate groups.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by Project U1301253 of NSFC and Project 201202032 of Liaoning Provincial Natural Science Foundation of China.

References

[1] M. A. Feki, F. Kawsar, M. Boussard, and L. Trappeniers, "The internet of things: the next technological revolution," *Computer*, vol. 46, no. 2, pp. 24–25, 2013.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] M. Li, Y. Liu, and L. Chen, "Nonthreshold-based event detection for 3D environment monitoring in sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 12, pp. 1699–1711, 2008.

[4] P. Brass, "Bounds on coverage and target detection capabilities for models of networks of mobile sensors," *ACM Transactions on Sensor Networks*, vol. 3, no. 2, article 9, Article ID 1240229, 2007.

[5] R. Tan, G. Xing, B. Liu, J. Wang, and X. Jia, "Exploiting data fusion to improve the coverage of wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 2, pp. 450–462, 2012.

[6] X. Wu, X. Zhu, W. Gong-Qing, and W. Ding, "Data mining with big data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 97–107, 2014.

[7] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 2, pp. 98–110, 1993.

[8] J.-S. Zhang and Y.-W. Leung, "Improved possibilistic C-means clustering algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 2, pp. 209–217, 2004.

[9] M.-S. Yang and C.-Y. Lai, "A robust automatic merging possibilistic clustering method," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 26–41, 2011.

[10] M. Barni, V. Cappellini, and A. Mecocci, "Comments on a possibilistic approach to clustering," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 3, pp. 393–396, 1996.

[11] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy C-means clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 4, pp. 517–530, 2005.

[12] Z. Xie, S. Wang, and F. L. Chung, "An enhanced possibilistic C-means clustering algorithm EPCM," *Soft Computing—A Fusion of Foundations, Methodologies and Applications*, vol. 12, no. 6, pp. 593–611, 2008.

[13] D. Li, H. Gu, and L. Zhang, "A fuzzy C-means clustering algorithm based on nearest-neighbor intervals for incomplete data," *Expert Systems with Applications*, vol. 37, no. 10, pp. 6942–6947, 2010.

[14] R. J. Hathaway and J. C. Bezdek, "Fuzzy C-means clustering of incomplete data," *IEEE Transactions on Systems, Man, and Cybernetics, B: Cybernetics*, vol. 31, no. 5, pp. 735–744, 2001.

[15] B. Liu, S. X. Xia, and Y. Zhou, "A sample-weighted possibilistic fuzzy clustering algorithm," *Acta Electronica Sinica*, vol. 40, no. 2, pp. 371–375, 2012.

[16] A. Schneider, "Weighted possibilistic C-means clustering algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 176–180, 2000.

[17] M. Filippone, F. Masulli, and S. Rovetta, "Applying the possibilistic C-means algorithm in kernel-induced spaces," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 3, pp. 572–584, 2010.

[18] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[19] Q. Zhang, Z. Chen, and L. T. Yang, "A nodes scheduling model based on Markov chain prediction for big streaming data analysis," *International Journal of Communication Systems*, 2014.

[20] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[21] W. Zhao, H. Ma, and Q. He, "Parallel k-means clustering based on mapreduce," in *Proceedings of the 1st International Conference on Cloud Computing*, pp. 674–679, Springer, Berlin, Germany, 2009.

[22] A. Ene, S. Im, and B. Moseley, "Fast clustering using MapReduce," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 681–689, ACM, New York, NY, USA, August 2011.

[23] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable K-means++," *Proceedings of the VLDB Endowment*, vol. 5, no. 7, pp. 622–633, 2012.

[24] T. Sun, C. Shu, F. Li, H. Yu, L. Ma, and Y. Fang, "An efficient hierarchical clustering method for large datasets with map-reduce," in *Proceedings of the International Conference on Parallel and Distributed Computing, Applications and Technologies, (PDCAT '09)*, pp. 494–499, Higashihiroshima, Japan, December 2009.

[25] H. Gao, J. Jiang, L. She, and Y. Fu, "A new agglomerative hierarchical clustering algorithm implementation based on the map reduce framework," *International Journal of Digital Content Technology and Its Applications*, vol. 4, no. 3, pp. 95–100, 2010.

[26] L. Li and Y. Xi, "Research on clustering algorithm and its parallelization strategy," in *Proceeding of the International Conference on Computational and Information Sciences (ICCIS '11)*, pp. 325–328, IEEE, Chengdu, China, October 2011.

[27] Y. He, H. Tan, W. Luo et al., "Mr-dbscan: an efficient parallel density-based clustering algorithm using MapReduce," in *Proceeding of the 17th IEEE International Conference on Parallel and Distributed Systems*, pp. 473–480, Tainan, Taiwan, December 2011.

[28] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[29] W. Lu, D. Chenyang, W. Baogang, S. Chunhui, and Y. Zhenchao, "Distributed affinity propagation clustering based on MapReduce," *Journal of Computer Research and Development*, vol. 49, no. 8, pp. 1762–1772, 2012.

[30] K. Yang, G. He, and G. He, "Research and application of MapReduce-based MST text clustering algorithm," in *Proceedings of the IEEE International Conference on Information Science and Technology (ICIST '12)*, pp. 753–757, IEEE, March 2012.

[31] Q. Yu and M. Dai, "Parallel fuzzy C-means algorithm based on MapReduce," *Computer Engineering and Applications*, vol. 49, no. 14, pp. 133–151, 2013.

[32] S. Papadimitriou and J. Sun, "DisCo: distributed Co-clustering with map-reduce: a case study towards petabyte-scale end-to-end mining," in *Proceeding of the 8th IEEE International Conference on Data Mining, (ICDM '08)*, pp. 512–521, IEEE, Pisa, Italy, December 2008.

[33] R. L. F. Cordeiro, C. Traina Jr., A. J. M. Traina, J. López, U. Kang, and C. Faloutsos, "Clustering very large multi-dimensional datasets with MapReduce," in *Proceeding of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 690–698, ACM, August 2011.

[34] L. Lu, Y. Gu, and R. Grossman, "dSimpleGraph: a novel distributed clustering algorithm for exploring very large scale unknown data sets," in *Proceeding of the10th IEEE International Conference on Data Mining Workshops (ICDMW '10)*, pp. 162–169, Washington, DC, USA, December 2010.

[35] A. Vergaraa, S. Vembua, T. Ayhanb, M. A. Ryanc, M. L. Homerc, and R. Huertaa, "Chemical gas sensor drift compensation using classifier ensembles," *Sensors and Actuators B: Chemical*, vol. 166-167, pp. 320–329, 2012.

[36] I. Rodriguez-Lujana, J. Fonollosaa, A. Vergarab, M. Homerc, and R. Huertaa, "On the calibration of sensor arrays for pattern recognition using the minimal number of experiments," *Chemometrics and Intelligent Laboratory Systems*, vol. 130, pp. 123–134, 2014.

[37] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy C-means algorithms for very large data," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, pp. 1130–1146, 2012.

[38] X. D. Han, Z. X. Xia, B. Liu, and Y. Zhou, "Kernel-based fast improved possibilistic C-means clustering method," *Computer Engineering and Applications*, vol. 47, no. 6, pp. 176–180, 2011.