Distributed Consensus-Based K-Means Algorithm in Switching Multi-Agent Networks*

LIN Peng · WANG Yinghui · QI Hongsheng · HONG Yiguang

DOI: 10.1007/s11424-018-7102-3
Received: 30 March 2017 / Revised: 3 July 2017

Abstract This paper discusses a distributed design for clustering based on the K-means algorithm in a switching multi-agent network, for the case when data are decentralized stored and unavailable to all agents. The authors propose a consensus-based algorithm in distributed case, that is, the double-clock consensus-based K-means algorithm (DCKA). With mild connectivity conditions, the authors show convergence of DCKA to guarantee a distributed solution to the clustering problem, even though the network topology is time-varying. Moreover, the authors provide experimental results on various clustering datasets to illustrate the effectiveness of the fully distributed algorithm DCKA, whose performance may be better than that of the centralized K-means algorithm.

Keywords Consensus-based algorithm, distributed K-means clustering, multi-agent network, switching topology.

1 Introduction

Due to the scalability, robustness, and low cost, distributed algorithms in multi-agent networks have attracted much attention in recent year. Each agent in the network can achieve the global goal based only on information exchange between neighbors or local measurement, because an agent has great difficulty getting all the information directly in a large-scale network. To guarantee all the agents to work for the same goal, the consensus-based algorithms have gained its popularity in the field of distributed control, estimation and optimization. In fact, network connectivity plays a vital role in achieving multi-agent coordination and making

LIN Peng · WANG Yinghui · QI Hongsheng · HONG Yiguang
Key Laboratory of Systems and Control, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, *China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing* 100190, *China.*
Email: penglin@amss.ac.cn; wangyinghuisdu@163.com; qihongsh@amss.ac.cn; yghong@iss.ac.cn.
*This research was supported by the National Key Research and Development Program of China under Grant No. 2016YFB0901902 and the National Natural Science Foundation of China under Grant Nos. 61573344, 61333001, 61733018, and 61374168.
◇ *This paper was recommended for publication by Editor HU Xiaoming.*

distributed algorithms/methods suitable or effective to deal with large data or complicated network structures. Due to energy-saving policies or link failures, variable topologies of communication or interaction networks are widely found in distributed designs, and it is no surprise that distributed consensus or optimization with time-varying topologies has become a hot topic[1–5].

In recent years, machine learning has attracted more and more research attention because of various applications in knowledge discovery, pattern recognition, and data mining[6, 7]. Data clustering, one of the unsupervised machine learning problems for data analysis, has also been significantly investigated and widely applied to many areas since it is proposed to effectively divide data into small subgroups[8]. Based on the similarity among data, partitional clustering algorithms (opposite to those hierarchical algorithms) become important research topics, partially because these algorithms can obtain all the clusters synchronously without requiring a hierarchical structure[9]. For instance, the K-means algorithm given in [10], one of the most popular partitional clustering algorithms, is well known for its simplicity and fast convergence rate[11, 12]. Some other clustering algorithms, such as the expectation-maximization (EM) algorithm[13] for Gaussian mixtures is also based on its idea.

Note that many clustering (including K-means) algorithms are basically centralized. However, the challenges with the rapid growth of data and large-scale networks, ask us to store and process data among different agents over the network. For example, in networked surveillance[14], wireless sensor networks[15–17], or distributed databases[18], data have been distributed over the network. To deal with these troubles, parallel K-means clustering schemes were studied widely. For example, [19] and [20] focused on multiprocessor architectures of parallel K-means algorithms. To realize parallel algorithms, people set a reliable center or the third party, to process all the information from the other agents over the network, and with job assignment, these parallel clustering designs can be applied to all the agents[21]. Moreover, privacy preservation has also been an important concern in the design of parallel clustering algorithms, [22] and [23] proposed different ways to partition data vertically and arbitrarily, before sharing the data. Nevertheless, if the central unit or the third party fails for some reasons, parallel clustering designs will be inoperative, too. Therefore, designing distributed clustering algorithms without requiring any center or third party becomes an urgent task for effective data processing or clustering.

Up to now, some distributed K-means clustering algorithms have been constructed. For example, [24] and [25] proposed distributed algorithms motivated by local and random sampling strategies, while [26] formulated a K-means clustering problem as a distributed optimization problem with consensus constraints on the centroids obtained by all agents, and provided a distributed algorithm based on the duality theory. [27] designed a distributed algorithm for vertically and horizontally distributed datasets. Additionally, [28] and [29] gave distributed K-means algorithms based on max-consensus and average-consensus. However, most existing distributed K-means algorithms were proposed for fixed topologies (which may fail in the cases of link failures or switching network connectivity) and analyzed on basis of simulations.

The motivation of this paper is to study the distributed K-means algorithm in a time-varying communication network with some convergence analysis. Here each agent in the network can have access to only a portion of clustering observations and can share information with its one-hop neighbours. The technical contribution of the paper includes:

1) Distributed design for switching networks: Different from many distributed K-means algorithms for fixed topologies[26, 28], we propose a double-clock consensus-based K-means algorithm (DCKA), based on time-varying networks. In other words, the topology of the network just need to be jointly-connected. DCKA is of relatively low communication cost, because it is unnecessary for the communication network to keep connected throughout.

2) Convergence and performance analysis: In light of graph theory[30], we prove that DCKA converges to a local optimal solution of the clustering problem. Different from existing distributed K-means algorithms[26] by mixing the consensus of the centroids of all the agents with the update of these centroids in one iteration step, DCKA separates these procedures in two clear steps, which can help us take good advantage of good properties such as high convergence rate of the conventional K-means algorithm[28]. Experimental results show that our algorithm is also less sensitive to initialization and therefore, is more likely to get better experimental results than conventional centralized ones.

The organization of the paper is presented as follows. First, some necessary preliminaries are introduced in Section 2. Next, the double-clock consensus-based K-means algorithm (DCKA) is proposed in time-varying networks and then analysed in Section 3. In Section 4, the experimental results are shown. Finally, the conclusions are presented in Section 5.

2 Preliminaries and Problem Formulation

In this section, we first introduce preliminaries about graph theory[30] for multi-agent systems. Then we give our problem formulation and briefly describe traditional centralized K-means algorithm.

2.1 Graph Theory

To describe networks in a distributed design, we consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as the information sharing topology among N agents. $\mathcal{V} = \{1, 2, \cdots, N\}$ represents the set of agents and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ denotes the set of the communication links among the agents. If agent i can receive information from agent h directly, then there exists a directed edge from h to i and denoted by $(h, i) \in \mathcal{E}$. Denote the one-hop neighbours of agent i as $\mathcal{N}_i = \{h | (h, i) \in \mathcal{E}\}$. The graph $\mathcal{G}$ is said to be undirected, if $(h, i) \in \mathcal{E}$, whenever $(i, h) \in \mathcal{E}$. A directed path, whose length is p, is a non-empty graph $\mathcal{P} = (\mathcal{V}_p, \mathcal{E}_p)$ of the form $\mathcal{V}_p = \{i_1, i_2, \cdots, i_{p+1}\} \subseteq \mathcal{V}$, $\mathcal{E}_p = \{(i_1, i_2), (i_2, i_3), \cdots, (i_p, i_{p+1})\} \subseteq \mathcal{E}_p$, where i_ks are all distinct. The graph $\mathcal{G}$ is strongly connected if there is a directed path between them for any pair $i, h \in \mathcal{V}$. More details can be found in [30].

Consider this graph $\mathcal{G}(s) = (\mathcal{V}, \mathcal{E}(s))$, which represents the time-varying communication topology among the agents at time s. The adjacency matrix of $\mathcal{G}(s)$ is denoted by $A(s) \in \mathcal{R}^{N \times N}$, whose elements are defined as follows:

(i) $a_{i,h}(s) > 0$ for any $(h, i) \in \mathcal{E}(s)$, including agent i itself, that is, $a_{i,i}(s) > 0$;

(ii) $a_{i,h}(s) = 0$ for any agent h that is not the one-hop neighbor of agent i.

Next, we consider two assumptions for an important time-varying network, which is the jointly-connected network. The first assumption is about the weighted adjacency matrix, and the second one is about the connectivity of the network.

Assumption 2.1 The weighted adjacency matrix $A(s)$ of graph $\mathcal{G}(s) = (\mathcal{V}, \mathcal{E}(s))$ is assumed to satisfy

(i) $A(s)$ is doubly stochastic;

(ii) For all $i \in \mathcal{V}$, $a_{i,i}(s) \geq \varepsilon$ and $a_{i,h}(s) \geq \varepsilon$ if $(h, i) \in \mathcal{E}(s)$, where ε is a positive scalar.

Assumption 2.2 The graph $\mathcal{G}(s) = (\mathcal{V}, \mathcal{E}(s))$ is jointly connected, i.e., the graph $(\mathcal{V}, \mathcal{E}(s) \cup \mathcal{E}(s+1) \cup \cdots \cup \mathcal{E}(s+\tau-1)$ is strongly connected for all $s \geq 0$ and some integer $\tau > 0$.

Clearly, Assumption 2.2 ensures that each agent i can get information from all its neighbors at least once during each period of τ, though the network topology is switching and may not be connected at each moment. The following result is also well known for this switching topology.

Lemma 2.3 (see [1]) *Under Assumptions* 2.1 *and* 2.2*, for all i, h and all $s_1 \geq s_2$, we have*

$$\left| [\varphi(s_1 : s_2)]_{i,h} - \frac{1}{N} \right| \leq \zeta^{-2} \varrho^{s_1 - s_2 + 1},$$

where $\zeta = 1 - \frac{\varepsilon}{4N^2}$, $\varrho = \zeta^{1/\tau}$, $\varphi(s_1 : s_2)$ is a transition matrix defined by $\varphi(s_1 : s_2) = A(s_2)A(s_2+1)\cdots A(s_1)$ with $\varphi(s_1 : s_1) = A(s_1)$.

2.2 Problem Formulation

For every agent $i \in \mathcal{V}$, a training set $Y_i = [y_i^1, y_i^2, \cdots, y_i^{m_i}] \in \mathcal{R}^{n \times m_i}$ is available, where n is the dimension of data and m_i is the size of data. Denote $Y = [Y_1, Y_2, \cdots, Y_N]$ as the whole dataset. The goal of agent i is to divide dataset Y_i into K clusters $[Y_i^1, Y_i^2, \cdots, Y_i^K]$, according to K-means clustering criterion. The K centroids, which agent i obtains to represent the clusters, may not belong to the dataset. Let c_k denote the k-th centroid and $C = \{c_1, c_2, \cdots, c_K\}$ be the collection of K centroids. The K-means clustering problem of agent i can be stated as follows:

$$\min_{\theta \in \mathcal{D}_i, C} \quad f_i(Y_i, C, \theta_i) = \sum_{j=1}^{m_i} \sum_{k=1}^{K} \theta_{i,k}^j ||y_i^j - c_k||^2, \tag{1}$$

where

$$\begin{aligned} \mathcal{D}_i = \Big\{ \theta_{i,k}^j : \sum_{k=1}^{K} \theta_{i,k}^j = 1,\ j = 1, 2, \cdots, m_i, \\ \theta_{i,k}^j = \{0, 1\},\ k = 1, 2, \cdots, K,\ j = 1, 2, \cdots, m_i \Big\} \end{aligned} \tag{2}$$

and

$$\theta_i = \{\theta_{ik}^j,\ k = 1, 2, \cdots, K,\ j = 1, 2, \cdots, m_i\}. \tag{3}$$

The similarity between the data points is measured by the Euclidean norm. In distributed

K-means clustering learning, the purpose of agents is to partition the dataset Y into K clusters $[Y^1, Y^2, \cdots, Y^K]$ and minimize the sum of squares of distances between the data points and the centroids of the clusters. Moreover, the K centroids that all agents get should be consistent with each other. The distributed K-means clustering problem, which was also introduced in [26, 28], is formulated as follows:

$$\min_{\theta\in\mathcal{D},C} \quad F(Y,C,\theta) = \sum_{i=1}^{N} f_i(Y_i, C, \theta_i), \tag{4}$$

where $\mathcal{D} = \{\mathcal{D}\}_{i=1}^N$ and $\theta = \{\theta_i\}_{i=1}^N$.

The cluster number K is assumed to be known for all agents as a priori knowledge in this paper; if it is unknown, the methods proposed in [31] can be used in our algorithms to estimate the number K.

Centralized K-means algorithm (CKA)[32] supplies a two-step (an assignment step and a refinement step) iterative approach to solve Problem (4). Denote $C(t) = \{c_1(t), c_2(t), \cdots, c_K(t)\}$ as the K centroids at time t. It starts with K random centroids $C(0)$ or through the K-means++ algorithm[33] with only one random centroid. In the assignment step, each data point y_i^j is assigned to the cluster which can be represented by the nearest centroid, that is,

$$\theta_{i,k}^j(t+1) = \arg\min_{\theta\in\mathcal{D}} \sum_{i=1}^{N}\sum_{j=1}^{m_i}\sum_{k=1}^{K} \theta_{i,k}^j ||y_i^j - c_k(t)||^2.$$

In other words,

$$\theta_{i,k}^j(t+1) = \begin{cases} 1, & \text{if } \ y_i^j \in \text{ Cluster } k; \\ 0, & \text{if } \ y_i^j \notin \text{ Cluster } k. \end{cases} \tag{5}$$

Obviously, $\sum_{i=1}^N \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1)$ and $\sum_{i=1}^N \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1) y_i^j$ stand for the size of data points and the summation of all the data points, respectively, which belong to the k-th cluster at time $t+1$. For simplicity, we denote

$$m_k(t+1) = \sum_{i=1}^{N}\sum_{j=1}^{m_i} \theta_{i,k}^j(t+1), \tag{6}$$

$$u_k(t+1) = \sum_{i=1}^{N}\sum_{j=1}^{m_i} \theta_{i,k}^j(t+1) y_i^j. \tag{7}$$

In the refinement step, the K centroids are updated by the new ones in each cluster, namely,

$$c_k(t+1) = \frac{\sum_{i=1}^N \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1) y_i^j}{\sum_{i=1}^N \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1)} = \frac{u_k(t+1)}{m_k(t+1)}, \quad k = 1, 2, \cdots, K. \tag{8}$$

The algorithm guarantees that a local optimal solution can be obtained. In CKA, both (5) and (8) need global (or centralized) information. Equation (5) requires the global information of centroids c_k and (8) asks for the global information of all the data points, which belong to cluster k. However, if data points are spatially distributed or collected by different agents, the centralized form may not work any more. It calls for the distributed implementation of the traditional K-means algorithms.

3 Distributed K-Means Algorithm

In this section, we first introduce our double-clock consensus-based K-means algorithm (DCKA). Then the convergence analysis, communication and computational complexity of DCKA are given.

3.1 Double-Clock Consensus-Based K-Means Algorithm

In this subsection, our double-clock consensus-based K-means algorithm (DCKA) is introduced, considering that the distributed coordination design for consensus or optimization was widely studied in jointly-connected cases[34].

To deal with (4), we define $C_i(t)$ as the set of centroids, which are obtained by agent i at time t. In the light of the centralized K-means clustering algorithm, given $C_i(t)$, $\theta_i(t+1)$ can be assigned by the following equation:

$$\theta_{i,k}^j(t+1) = \arg\min_{\theta_i \in \mathcal{D}_i} \sum_{j=1}^{m_i} \sum_{k=1}^{K} \theta_{i,k}^j ||y_i^j - c_{i,k}(t)||^2.$$

In another word,

$$\theta_{i,k}^j(t+1) = \begin{cases} 1, & \text{if } y_i^j \in \text{ Cluster } k; \\ 0, & \text{if } y_i^j \notin \text{ Cluster } k. \end{cases} \tag{9}$$

It is obvious that $\sum_{j=1}^{m_i} \theta_{i,k}^j(t+1)$ stands for the size of data, in the k-th cluster at time $t+1$, which belong to agent i. We denote

$$m_{i,k}(t+1) = \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1), \tag{10}$$

and $M_i(t+1)$ as the collection of $m_{i,1}(t+1)$. Then a new local summation $u_{i,k}(t+1)$ can be obtained by

$$u_{i,k}(t+1) = \sum_{j=1}^{m_i} \theta_{i,k}^j(t+1) y_i^j. \tag{11}$$

Hence, the global centroid $c_k(t+1)$ in (8) is rewritten as follows:

$$c_k(t+1) = \frac{\sum_{i=1}^{N} u_{i,k}(t+1)}{\sum_{i=1}^{N} m_{i,k}(t+1)}, \quad k = 1, 2, \cdots, K. \tag{12}$$

Denote $x_{i,k}(0,t)=\mathrm{col}\{x^1_{i,k}(0,t),x^2_{i,k}(0,t)\}$, where

$$\begin{cases}x^1_{i,k}(0,t)=u_{i,k}(t),\\ x^2_{i,k}(0,t)=m_{i,k}(t).\end{cases} \tag{13}$$

In each iteration of the algorithm, agent i updates its centroids through three steps: A local K-means step, a consensus step over the jointly-connected topology, and a local update step:

1) Local K-means step: Agent i calculates $\theta^j_{i,k}(t+1)$, $m_{i,k}(t+1)$ and $u_{i,k}(t+1)$ according to (9), (10), and (11), respectively.

2) Consensus step: Over an information sharing topology $\mathcal{G}(s)=(\mathcal{V},\mathcal{E}(s))$, which is time-varying and satisfies Assumptions 2.1 and 2.2, agent i exchanges information with its one-hop neighbor h according to the following equation until consensus:

$$x_{i,k}(s+1,t+1)=\sum_{h\in\mathcal{N}_i(s)}a_{i,h}(s)x_{h,k}(s,t+1),\quad k=1,2,\cdots,K. \tag{14}$$

The stop time of the consensus step over jointly-connected topology is denoted by S.

3) Local update step: Agent i updates $c_{i,k}(t+1)$ as follows:

$$c_{i,k}(t+1)=\frac{x^1_{i,k}(S,t+1)}{x^2_{i,k}(S,t+1)},\quad k=1,2,\cdots,K.$$

Let us summarize the procedure of DCKA in Algorithm 1.

Algorithm 1: **Double-clock consensus-based K-means algorithm (DCKA)**

```
Input The datasets $Y=[Y_1,Y_2,\cdots,Y_N]$ and $\{C_i(0)\}_{i=1}^N$
1: for $t=0,1,\cdots$ do
2:     for $i=1,2,\cdots,N$ do
3:         Calculate $\theta^j_{i,k}(t+1)$ according to (9).
4:         Calculate $m_{i,k}(t+1)$ according to (10).
5:         Calculate $u_{i,k}(t+1)$ according to (11).
6:     end for
7:     for $i=1,2,\cdots,N$ do
8:         Calculate $x_{i,k}(0,t+1)$ according to (13)
9:     end for
10:    for $i=1,2,\cdots,N$ do
11:        Do consensus step until convergence according to (14), obtain $x_{i,k}(S,t+1)$.
12:    end for
13:    for $i=1,2,\cdots,N$ do
14:        $c_{i,k}(t+1)=\frac{x^1_{i,k}(S,t+1)}{x^2_{i,k}(S,t+1)}$.
15:    end for
16: end for
```

Remark 3.1 Different from the existing works, DCKA proposed in this paper has two clocks: A clock of the of the consensus between agents over time axis s and another clock of the local K-means computation over time axis t. In the consensus step, agent i exchanges information with its one-hop neighbor agent h, until the consensus can be achieved within some tolerant bound, which is easy to implement for fixed time clock $t+1$ of the local steps and can be adapted to the case where the global communication topology is unavailable. The consensus rate in the consensus step is still exponential, even if the network topologies are jointly-connected switching due to link failure/recovery procedures or energy saving policies.

3.2 Analysis of DCKA

In this subsection, the convergence analysis is given first and then the communication and computational complexity of DCKA are discussed.

Theorem 3.2 is provided to address the convergence of DCKA.

Theorem 3.2 *Under Assumptions* 2.1 *and* 2.2*, DCKA achieves a local optimal solution of distributed clustering problem* (4)*.*

Proof Denote $\overline{x}_k(0,t+1)=\mathrm{col}\{x_k^1(0,t+1),x_k^2(0,t+1)\}$, where

$$\begin{cases}\overline{x}_k^1(0,t+1)=\dfrac{1}{N}\displaystyle\sum_{i=1}^N x_{i,k}^1(0,t+1)=\dfrac{\sum_{i=1}^N u_{i,k}(t+1)}{N},\\ \overline{x}_k^2(0,t+1)=\dfrac{1}{N}\displaystyle\sum_{i=1}^N x_{i,k}^2(0,t+1)=\dfrac{\sum_{i=1}^N m_{i,k}(t+1)}{N}.\end{cases}$$

From the definition of $c_k(t+1)$ in (12), we have $c_k(t+1)=\frac{\overline{x}_k^1(0,t+1)}{\overline{x}_k^2(0,t+1)}$. Define $\widehat{c}_{i,k}(s,t+1)$ as follows:

$$\widehat{c}_{i,k}(s,t+1)=\frac{x_{i,k}^1(s,t+1)}{x_{i,k}^2(s,t+1)},\quad k=1,2,\cdots,K, \tag{15}$$

which is the estimation of agent i for $c_k(t+1)$ at time s in the consensus step.

In light of the fact that CKA can obtain a local optimal solution, the conclusion follows if we can show that $\widehat{c}_{i,k}(s,t+1)$ converge to $c_k(t+1)$ as $s\to\infty$ first, and then prove there exists an S, such that the error between $\widehat{c}_{i,k}(S,t+1)$ and $c_k(t+1)$ have no effect on $\theta_{i,k}^j(t+2)$.

First, we prove that $\widehat{c}_{i,k}(s,t+1)$ converge to $c_k(t+1)$ as $s\to\infty$.

For any s with $s\geq 0$, we have

$$x_{i,k}^1(s,t+1)=\sum_{h=1}^N[\varphi(s-1:0)]_{i,h}x_{h,k}^1(0,t+1), \tag{16}$$

$$x_{i,k}^2(s,t+1)=\sum_{h=1}^N[\varphi(s-1:0)]_{i,h}x_{h,k}^2(0,t+1). \tag{17}$$

Let

$$\xi^1_{i,k}(s,t+1) = x^1_{i,k}(s,t+1) - \overline{x}^1_k(0,t+1) = \sum_{h=1}^{N}\left([\varphi(s-1:0)]_{i,h} - \frac{1}{N}\right)x^1_{h,k}(0,t+1),$$
$$\xi^2_{i,k}(s,t+1) = x^2_{i,k}(s,t+1) - \overline{x}^2_k(0,t+1) = \sum_{h=1}^{N}\left([\varphi(s-1:0)]_{i,h} - \frac{1}{N}\right)x^2_{h,k}(0,t+1).$$

By Lemma 2.3, we obtain

$$\begin{aligned}\|\xi^1_{i,k}(s,t+1)\| &\le \sum_{h=1}^{N}\left|[\varphi(s-1:0)]_{i,h} - \frac{1}{N}\right| \|x^1_{h,k}(0,t+1)\| \\ &\le N\zeta^{-2}\varrho^s \max_h \|x^1_{h,k}(0,t+1)\|, \qquad (18)\\ \|\xi^2_{i,k}(s,t+1)\| &\le \sum_{h=1}^{N}\left|[\varphi(s-1:0)]_{i,h} - \frac{1}{N}\right| \|x^2_{h,k}(0,t+1)\| \\ &\le N\zeta^{-2}\varrho^s \max_h \|x^2_{h,k}(0,t+1)\|. \qquad (19)\end{aligned}$$

Therefore,

$$\begin{aligned}&\|\widehat{c}_{i,k}(s,t+1) - c_k(t+1)\| \\ =&\left\|\frac{x^1_{i,k}(s,t+1)}{x^2_{i,k}(s,t+1)} - \frac{\overline{x}^1_k(0,t+1)}{\overline{x}^2_k(0,t+1)}\right\| \\ =&\left\|\frac{\overline{x}^1_k(0,t+1) + \xi^1_{i,k}(s,t+1)}{\overline{x}^2_k(0,t+1) + \xi^2_{i,k}(s,t+1)} - \frac{\overline{x}^1_k(0,t+1)}{\overline{x}^2_k(0,t+1)}\right\| \\ =&\left\|\frac{\xi^1_{i,k}(s,t+1)\overline{x}^2_k(0,t+1) - \xi^2_{i,k}(s,t+1)\overline{x}^1_k(0,t+1)}{[\overline{x}^2_k(0,t+1) + \xi^2_{i,k}(s,t+1)]\overline{x}^2_k(0,t+1)}\right\| \\ \le&\frac{\left\|\xi^1_{i,k}(s,t+1)\right\| \left\|\overline{x}^2_k(0,t+1)\right\| + \left\|\xi^2_{i,k}(s,t+1)\right\| \left\|\overline{x}^1_k(0,t+1)\right\|}{\left\|\overline{x}^2_k(0,t+1)\right\|^2}. \qquad (20)\end{aligned}$$

Denote

$$\sigma_k(t+1) = \max\left\{\frac{\max_h\|x^1_{h,k}(0,t+1)\|}{\|\overline{x}^2_k(0,t+1)\|}, \frac{\|\overline{x}^1_k(0,t+1)\|\max_h\|x^2_{h,k}(0,t+1)\|}{\|\overline{x}^2_k(0,t+1)\|^2}\right\}.$$

According to (18), we get

$$\|\widehat{c}_{i,k}(s,t+1) - c_k(t+1)\| \le 2N\zeta^{-2}\varrho^s\sigma_k(t+1).$$

Hence, $\forall\delta > 0$, there exists an $S = \log_\rho(\frac{\zeta^2\delta}{2N\sigma_k(t+1)})$, for any $s \geq S$,

$$\|\widehat{c}_{i,k}(s,t+1) - c_k(t+1)\| \leq \delta,$$

where $\zeta = 1 - \frac{\varepsilon}{4N^2}$, $\varrho = \zeta^{1/\tau}$. In other words, $\widehat{c}_{i,k}(s,t+1)$ converge to $c_k(t+1)$ as $s \to \infty$.

Then we show that there exists an S, such that the error between $\widehat{c}_{i,k}(S,t+1)$ and $c_k(t+1)$ have no effect on $\theta^j_{i,k}(t+2)$.

Since θ is decided by the centroids of each cluster, we denote $\theta(t+1) = \theta[C(t)]$. Given the centroids $C(t+1)$, we take

$$\delta(t+1) = \min_{1\leq i\leq N}\min_{1\leq k\leq K}\delta_{i,k}(t+1), \tag{21}$$

where

$$\delta_{i,k}(t+1) = \min_{y\in Y_i^k}\min_{k'\neq k}\left\{\frac{1}{2}(\|y - c_{k'}(t+1)\|^2 - \|y - c_k(t+1)\|^2)\right\}.$$

Then for any $c_{i,k} \in \mathbb{B}(c_k(t+1), \delta(t+1))$, we have $\theta[\{C_i\}_{i=1}^N] = \theta[C(t+1)]$, where $C_i = \{c_{i,k}\}_{k=1}^K$.

For this $\delta(t+1)$, if we choose $S \geq \log_\rho(\frac{\zeta^2\delta(t+1)}{2N\sigma(t+1)})$, where

$$\sigma(t+1) = \max_k \sigma_k(t+1),$$

then $\widehat{c}_{i,k}(S,t+1) \in \mathbb{B}(c_k(t+1), \delta(t+1))$, for all $i = 1, 2, \cdots, N$ and $k = 1, 2, \cdots, K$.

In other words, if $S \geq \log_\rho(\frac{\zeta^2\delta(t+1)}{2N\sigma(t+1)})$, we have $\theta[\{C_i(t+1)\}_{i=1}^N] = \theta[C(t+1)]$. Therefore, DCKA and CKA should get the same $\theta^j_{i,k}(t+2)$ in the next iteration. ∎

Next, we analyse the communication and computational complexity of DCKA. Denote T as the maximal iteration of DCKA and $M = \sum_{i=1}^N m_i$ as the total number of the data points in all agents. Denote N_l as the average number of links in the communication network. When the topology of the communication network is assumed to be jointly-connected, N_l can be much smaller than that when the network keeps connected throughout.

Then we provide another main result.

Theorem 3.3 *The communication consumption, space complexity, and time complexity of DCKA are $O(ST)$, $O((2N+M)K)$, and $O(TK(M+NS+N))$, respectively.*

Proof Let us first check the communication complexity of DCKA, which is close related to the clusters, iterations, network links, etc. During one loop of the consensus step in DCKA, each agent sends $x_{i,k}(s,t)$ about cluster k to its one-hop neighbors. By denoting N_b as the number of bytes for sending $x_{i,k}(s,t)$, the communication consumption is $2N_lKSN_b$ during S loops. Hence, the total communication consumption is $2N_lKSTN_b \sim O(ST)$.

Then we check the computational complexity of DCKA. Agents in the network need $O(NK)$ space to store K centroids, $O(MK)$ space to store the value of θ, and $O(NK)$ space in consensus step. Hence, the total space complexity is $O((2N+M)K)$. The time complexity of DCKA depends on the size of dataset, the number of agents in the network, number of clusters and related factors. Per iteration, the time complexity is $O(KM)$ for the local K-means step, $O(NKS)$ for the consensus step, $O(NK)$ for the local update step. Hence, the total time complexity of DCKA is $O(TK(M + NS + N))$. Moreover, when the size of dataset Y is very large, the total time complexity of DCKA is $O(TKM)$. ▮

In DCKA, agents in the considered network share their information with their one-hop neighbors to get the same K centroids. Since the consensus of all the centroids in all agents and the update of the centroids are separated completely, DCKA inherits the convergent rate of the CKA. The communication topology among the agents is assumed to be jointly-connected. Every agent in the network communicates with its one-hop neighbors at least once during each period of τ. The jointly-connected connectivity is applicable to failures of the links among the agents and also the reduction of the communication cost. Different from the results given in [24], we give convergence analysis for switching-network cases due to the link switching (maybe resulting from link failure or active energy saving).

4 Experimental Results

In this section, examples are given to evaluate the performance of our consensus-based clustering algorithms DCKA on both synthetic datasets and open dataset from [35]. The first example is on synthetic dataset to illustrate the performance of our algorithm on switching communication networks, and the results that DCKA gets are as well as the CKA. Compared with the DKM algorithm proposed in [26], which can not adapt to the time-varying communication network, DCKA converges faster and need less communication cost. The second one is on the open datasets to reveal that DCKA is less sensitive to the initialization compared with CKA, and more likely to get better results.

Example 4.1 The example is on an synthetic dataset, wherein the clustering data correspond to $K = 9$ different Gaussian distributions in a two-dimension space. The nine different Gaussian distributions are assumed to have different means, which are denoted by $w_1 = (0, 0)^{\mathrm{T}}$, $w_2 = (0, 3)^{\mathrm{T}}$, $w_3 = (0, -3)^{\mathrm{T}}$, $w_4 = (3, 0)^{\mathrm{T}}$, $w_5 = (-3, 0)^{\mathrm{T}}$, $w_6 = (3, -3)^{\mathrm{T}}$, $w_7 = (3, 3)^{\mathrm{T}}$, $w_8 = (-3, 3)^{\mathrm{T}}$ and $w_9 = (-3, -3)^{\mathrm{T}}$, respectively, and they share the same covariance matrix $\Sigma = 0.64I_2$, where I_2 is a two-dimension identity matrix. Suppose that there are five agents in the network. Each agent i has access to a partial dataset Y_i of size 900, with 100 data points per Gaussian distribution which are all randomly generated.

The goal of each agent i is to find the same nine cluster centroids with a partial dataset Y_i. To measure the clustering performance of the proposed algorithms, the sum of squares of distances (SSD) is defined as follows [cf. (4)]:

$$\mathrm{SSD} = F(Y, C_Y) = \sum_{i=1}^{N} \sum_{j=1}^{m_i} \sum_{k=1}^{K} \theta_{i,k}^{j} ||y_i^j - c_{i,k}||^2,$$

where $C_Y = \{c_{i,k}\}$ and $Y = \cup_{i=1}^{5} Y_i$. The initial cluster centroids $c_{i,k}(0), k = 1, 2 \cdots, 9$, for agent i are randomly chosen from the partial dataset Y_i.

In DCKA, the communication topology of is assumed to be jointly-connected. The communication topology is assumed to be Figure 1(b) at time $s = 2n$, and Figure 1(c) at time $s = 2n + 1$, for $n = 1, 2, \cdots$.

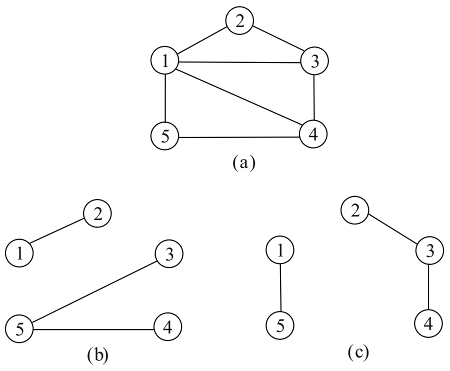

Figure 1 The topologies of the networks: (a) is connected; (b) with (c) is jointly-connected

First, we investigate the SSD of DCKA compared with CKA, and the results are presented in Figure 2. The SSDs are the same for both of the algorithms, which is 5028.54. Figure 3 displays the clustering results more visually, with centroids $c_1 = (0.0946, 0.0449)^{\mathrm{T}}$, $c_2 = (-0.0336, 3.0507)^{\mathrm{T}}$, $c_3 = (-0.024, -3.0223)^{\mathrm{T}}$, $c_4 = (2.9454, -0.0521)^{\mathrm{T}}$, $c_5 = (-3.0908, 0.013)^{\mathrm{T}}$, $c_6 = (3.0551, -3.058)^{\mathrm{T}}$, $c_7 = (3.0144, 3.0049)^{\mathrm{T}}$, $c_8 = (-2.9562, 3.0391)^{\mathrm{T}}$. The pentagonal ones in Figure 3 are centroids obtained by CKA. The other ones represent the centroids obtained by DCKA. Figure 3 illustrates that DCKA performs as well as CKA, in spite of switching communication topology among the agents.

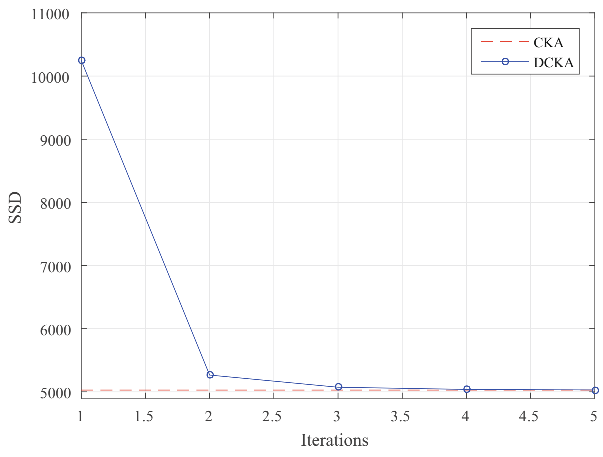

Figure 2 The sum of squares of distances (SSDs) of CKA and DCKA

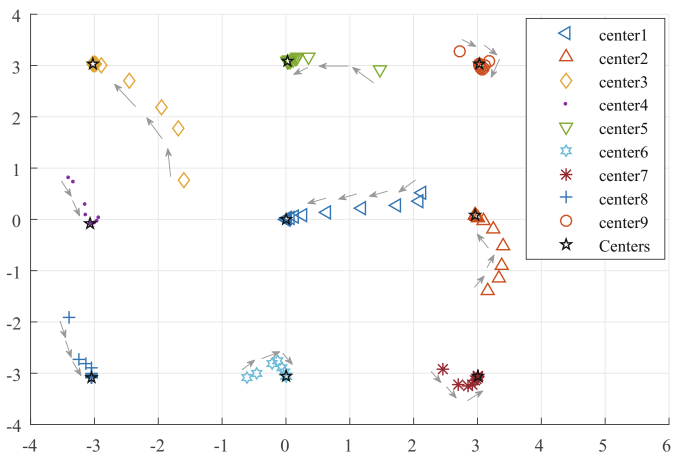

Figure 3 The centroids obtained by different algorithms: Pentagonal ones are for CKA; the other ones for DCKA

Next, we investigate the consensus performance of the proposed algorithm DCKA. Figure 4 shows the performance $x_{i,1}^2(s,t)$ of each agent i for DCKA. We can see that $x_{i,1}^2(s,t)$ converges fast to the value $\overline{x}_1^2(0,t) = \frac{\sum_{i=1}^N m_{i1}(t)}{N}$ in consensus step, which the rate of convergence is, in fact, exponential.

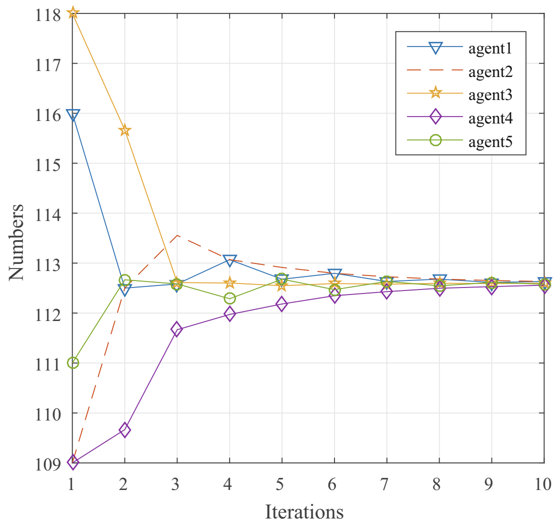

Figure 4 The consensus of $x_i^2(t)$ for all $i = 1, 2, \cdots, N$

The behavior of DCKA with different values of K should also be taken into consideration. Figure 5 shows that DCKA performs as well as CKA in spite of different values of K.

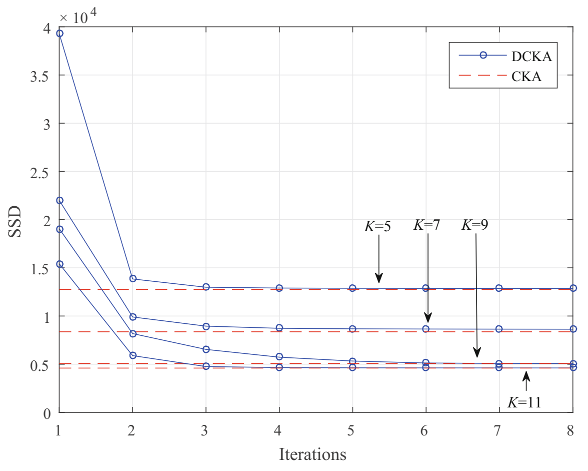

Figure 5 The performance of distributed algorithms for different values of K

Since the clustering results of DCKA and CKA are both related to the choice of the initial centroids, we simulated 100 Monte Carlo runs with different Ks in order to test the algorithms. The results of which are shown in Table 1. From Table 1, we can see that both DCKA and CKA share the same minimum SSD. However, DCKA outperforms CKA on the means and the standard deviation of SSDs thanks to the diversity of initial centroids of DCKA. CKA has to be initialized with K centroids randomly, while each agent is initialized with K centroids in DCKA according to their partial datasets $Y_i, i = 1, 2, \cdots, 5$. Thus, DCKA can usually get better results than CKA.

Table 1 Results about 100 Monte Carlo runs on different Ks

K	CKA			DCKA		
	Min.	Mean	Std. Dev.	Min.	Mean	Std. Dev.
5	12688.36	12902.44	89.75	12688.36	12835.69	75.32
7	8353.62	8518.41	101.65	8353.62	8476.12	77.07
9	5028.54	5156.54	130.14	5028.54	5121.97	110.18
11	4496.72	4544.88	21.84	4496.72	4534.21	19.39

Finally, we investigate the communication and computation consumption of DCKA compared to the DKM algorithm proposed in [26]. With the communication topology chosen as Figure 1(a), the DKM algorithm can also be applied to solve the clustering problem. The SSD performance is shown in Figure 6, where $K = 9$ and the parameter in DKM is assumed to be $\eta = 10$. In DKM, about 30 iterations are needed to complete the clustering task, which converges slowly. The agents need to communicate with their neighbors about 30 times with total communication consumption $210N_b$. However, the communication cost of DCKA is only about $150N_b$, which is much lower than that of DKM. Still, agents of the DKM algorithm in the network process all the 4500 data points about 30 times, while agents of DCKA only need to process the whole dataset about 5 times.

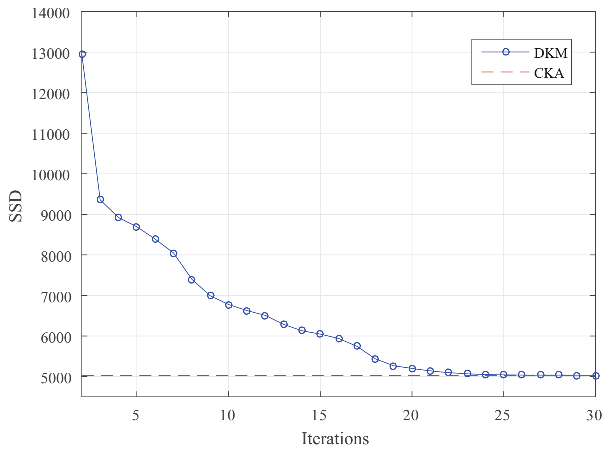

Figure 6 The performance of DKM, the communication topology is assumed to be connected; $K = 9$, $\eta = 10$

Example 4.2 Here, we choose Birch1, Birch2, and Birch3 to be the experiment datasets. Each of them consists of 100000 entities which are all two-dimension vectors, and the true cluster numbers are all $K = 100$. Consider a 20-agent network. Each agent can get 5000 data points randomly to form its own partial dataset Y_i, while CKA is taken on the entire dataset $Y = \{Y_i\}_{i=1}^N$.

In DCKA, the time-varying communication topology of the agents is assumed to be jointly-connected. At time $s = 3n$, $s = 3n + 1$ and $s = 3n + 2, n = 1, 2, \cdots$ the topologies are given in Figure 7(b), Figure 7(c), and Figure 7(d), respectively, and Figure 7(a) is the union of them.

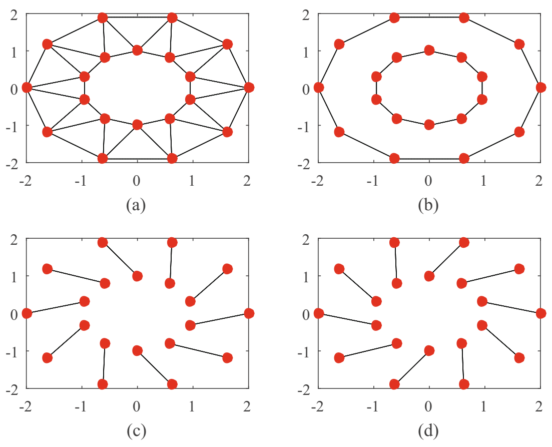

Figure 7 The topologies of the networks, which consists of 20 agents

We simulated 60 Monte Carlo runs with different Ks in order to test the SSD performance of the two algorithms on these datasets, the results of which are shown in Table 2, Table 3, and Table 4, respectively. Among all three datasets, DCKA outperforms CKA on the means and the standard deviation of SSD, and CKA only outperforms DCKA on the minimum of SSD on Birch3, when $K = 80$ and 100. The results reveal that DCKA is less sensitive to the initialization compared with CKA, and can usually get good results.

Table 2 Results about 60 Monte Carlo runs on Birch1 ($\times 10^{13}$)

K	CKA			DCKA		
	Min.	Mean	Std. Dev.	Min.	Mean	Std. Dev.
60	22.608	22.995	0.114	22.608	22.842	0.118
80	15.091	15.376	0.202	15.091	15.339	0.155
100	9.696	10.425	0.363	9.672	10.354	0.306

Table 3 Results about 60 Monte Carlo runs on Birch2 ($\times 10^{11}$)

K	CKA			DCKA		
	Min.	Mean	Std. Dev.	Min.	Mean	Std. Dev.
60	34.206	44.336	2.242	30.195	31.615	0.926
80	16.348	17.777	0.927	15.434	16.091	0.365
100	6.332	8.607	1.086	4.718	6.275	0.653

Table 4 Results about 60 Monte Carlo runs on Birch3 ($\times 10^{13}$)

K	CKA			DCKA		
	Min.	Mean	Std. Dev.	Min.	Mean	Std. Dev.
60	7.068	7.907	0.366	7.068	7.761	0.251
80	4.945	5.271	0.188	4.955	5.061	0.184
100	3.875	4.071	0.124	3.876	4.058	0.105

5 Conclusion

In this paper, we discussed the K-means clustering of a multi-agent network and provide a fully distributed algorithm for the cases with data stored in different agents or unavailable to all agents. We proposed a double-clock consensus-based algorithm DCKA to solve the clustering problem by making agents achieve consensus without the global information over a jointly-connected topology. Moreover, we gave the convergence analysis for the algorithm, and also provided examples by using various real clustering datasets in order to demonstrate the effectiveness of the proposed distributed algorithm.

References

[1] Nedic A and Ozdaglar A, Distributed subgradient methods for multi-agent optimization, *IEEE Trans. Automatic Control*, 2009, **54**(1): 48–61.

[2] Lou Y C, Hong Y G, and Shi G D, Target aggregation of second-order multi-agent systems with switching interconnection, *Journal of Systems Science and Complexity*, 2012, **25**(3): 430–440.

[3] Yi P and Hong Y G, Stochastic sub-gradient algorithm for distributed optimization with random sleep scheme, *Control Theory and Technology*, 2015, **13**(4): 333–347.

[4] Lou Y C, Hong Y G, and Wang S Y, Distributed continuous-time approximate projection protocols for shortest distance optimization problems, *Automatica*, 2016, **69**: 289–297.

[5] Liu X Y, Sun J, Dou L H, et al., Leader-following consensus for discrete-time multi-agent systems with parameter uncertainties based on the event-triggered strategy, *Journal of Systems Science and Complexity*, 2017, **30**(1): 30–45.

[6] Fayyad U M, Piatetsky-Shapiro G, Smyth P, et al., *Advances in Knowledge Discovery and Data Mining*, AAAI Press, Menlo Park, California, 1996.

[7] Basu S, Davidson I, and Wagstaff K, *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, CRC Press, Boca Raton, USA, 2008.

[8] Jain A K and Dubes R C, *Algorithms for Clustering Data*, Prentice-Hall, Inc., 1988.

[9] Jain A K, Data clustering: 50 years beyond K-means, *Pattern Recognition Letters*, 2010, **31**(8): 651–666.

[10] Lloyd S, Least squares quantization in PCM, *IEEE Trans. Information Theory*, 1982, **28**(2): 129–137.

[11] Bottou L and Bengio Y, Convergence properties of the K-means algorithms, *Advances in Neural Information Processing Systems*, 1995, **25**(2): 585–592.

[12] Ostrovsky R, Rabani Y, Schulman L J, et al., The effectiveness of lloyd-type methods for the K-means problem, *Proceedings of the* 47*th Annual IEEE Symposium on Foundations of Computer Science*, Berkeley, 2006.

[13] Gu D B, Distributed em algorithm for gaussian mixtures in sensor networks, *IEEE Transactions on Neural Networks*, 2008, **19**(7): 1154–1166.

[14] Greenhill S and Venkatesh S, Distributed query processing for mobile surveillance, *Proceedings of the* 15*th ACM International Conference on Multimedia*, Augsburg, 2007.

[15] Considine J, Li F F, Kollios G, et al., Approximate aggregation techniques for sensor databases, *Proceedings of the* 20*th Conference on Data Engineering*, Boston, 2004.

[16] Greenwald M B and Khanna S, Power-conserving computation of order-statistics over sensor networks, *Proceedings of the* 23*rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Madison, 2004.

[17] Lewis F L, *Wireless Sensor Networks*, John Wiley & Sons, Inc., 2005.

[18] Corbett J C, Dean J, Epstein M, et al., Spanner: Google's globally distributed database, *ACM Tran. Computer Systems*, 2013, **31**(3): 251–264.

[19] Joshi M N, Parallel K-means algorithm on distributed memory multiprocessors, *Computer*, 2003, **9**: 3–15.

[20] Dhillon I S and Modha D S, A data-clustering algorithm on distributed memory multiprocessors,

Large-Scale Parallel Data Mining, Springer, 2002, 245–260.

[21] Hajiee M, A new distributed clustering algorithm based on K-means algorithm, *Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering*, Chengdu, 2010.

[22] Vaidya J and Clifton C, Privacy-preserving K-means clustering over vertically partitioned data, *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, 2003.

[23] Jagannathan G and Wright R N, Privacy-preserving distributed K-means clustering over arbitrarily partitioned data, *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Chicago, 2005.

[24] Datta S, Giannella C, and Kargupta H, K-means clustering over a large, dynamic network, *Proceedings of the 2006 SIAM International Conference on Data Mining*, Bethesda, 2006.

[25] Datta S, Giannella C, and Kargupta H, Approximate distributed K-means clustering over a peer-to-peer network, *IEEE Trans. Knowledge and Data Engineering*, 2009, **21**(10): 1372–1388.

[26] Forero P A, Cano A, and Giannakis G B, Distributed clustering using wireless sensor networks, *IEEE Journal of Selected Topics in Signal Processing*, 2011, **5**(4): 707–724.

[27] Khedr A M and Bhatnagar R K, New algorithm for clustering distributed data using K-means, *Computing & Informatics*, 2014, **33**(4): 943–964.

[28] Oliva G, Setola R, and Hadjicostis C N, Distributed K-means algorithm, arXiv:1312.4176, 2013.

[29] Liu Q H, Fu W M, Qin J H, et al., Distributed K-means algorithm for sensor networks based on multi-agent consensus theory, *Proceedings of 2016 IEEE International Conference on Industrial Technology*, Taipei, China, 2016.

[30] West D B, *Introduction to Graph Theory*, 2nd Edition, Prentice Hall, Inc. Upper Saddle River, 2001.

[31] Forero P A, Cano A, and Giannakis G B, Distributed feature-based modulation classification using wireless sensor networks, *Proceedings of IEEE Military Communications Conference*, San Diego, 2008.

[32] Hartigan J A and Wong M A, Algorithm as 136: A K-means clustering algorithm, *Applied Statistics*, 1979, **28**(1): 100–108.

[33] Arthur D and Vassilvitskii S, K-means++: The advantages of careful seeding, *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, 2007.

[34] Yuan D M, Ho D W, and Xu S Y, Zeroth-order method for distributed optimization with approximate projections, *IEEE Trans. Neural Networks and Learning Systems*, 2016, **27**(2): 284–294.

[35] Franti P, et al., Clustering datasets, 2015, http://cs.uef.fi/sipu/datasets/.