2022 IEEE 61st Conference on Decision and Control (CDC)
December 6-9, 2022. Cancún, Mexico

Distributed Finite-Time k-means Clustering with Quantized Communucation and Transmission Stopping

Apostolos I. Rikos, Gabriele Oliva, Christoforos N. Hadjicostis, and Karl H. Johansson

2022 IEEE 61st Conference on Decision and Control (CDC) | 978-1-6654-6761-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CDC51059.2022.9992699

***Abstract*— In this paper, we present a distributed version of the k-means algorithm for multi-agent systems with directed communication links. The goal of k-means is to partition the network's agents in mutually exclusive sets (groups) such that agents in the same set have (and possibly share) similar information and are able to calculate a representative value for their group. Our distributed algorithm allows each node to transmit quantized values in an event-driven fashion, and exhibits distributed stopping capabilities. Transmitting quantized values leads to more efficient usage of the available bandwidth and reduces the communication bottleneck, whereas distributed stopping preserves available resources. We characterize the properties of the proposed distributed algorithm and show that its execution (on any static and strongly connected digraph) will partition all agents in mutually exclusive clusters in finite time. We conclude with examples that illustrate the operation, performance, and potential advantages of the proposed algorithm.**

I. Introduction

Data clustering is a fundamental problem whereby data is partitioned in groups and a representative value is identified for each group. Such methods are adopted in a broad variety of different applications, ranging from customer segmentation [1] to cybersecurity [2]. Notably, in the case of wireless sensor networks, a large amount of data is typically generated or sensed [3]. In this view, the ability of a set of agents to collectively cluster their sensed data would allow them to contain the overall amount of information and to establish functional connections among the agents, e.g., by identifying other agents with similar values. Establishing functional connections among agents over a wireless sensor network can enable interesting applications, such as coverage control, where a network of mobile agents is in charge of covering a finite set of points of interest [4].

In the literature, there have been various works on distributed clustering (e.g., see [5], [6] and references therein) and recently on distributed algorithms such as k-means clustering [7]–[9] and C-means clustering [10]. However, most clustering algorithms feature message exchanges consisting of floating point values, which leads to a significant increase in computational and bandwidth requirements, and may be responsible for introducing quantization errors or approximations.

Main Contributions. In this paper, we aim to analyze the distributed k-means clustering problem while we reduce the communication bottleneck between nodes. We focus on the realistic scenario where a set of wireless sensor nodes, which are spread over an area, collect data from the environment and communicate with quantized messages. We present a distributed algorithm which operates in an event-triggered fashion and solves the k-means clustering problem in finite time. Furthermore, in order to preserve available resources, nodes are able to determine whether the algorithm converged so as to terminate their operation. The main contributions of our paper are the following.

A. We present a novel distributed algorithm for solving the k-means clustering problem. During its operation, nodes exchange quantized messages of finite length with their neighboring nodes; see Algorithm 2.

B. We show that our proposed algorithm converges in a deterministic manner after a finite number of time steps. We calculate an explicit deterministic upper bound on the required time steps for the convergence of our algorithm. Our bound depends on the network structure and the number of centroid calculations; see Theorem 2.

Current literature comprises centralized or distributed algorithms whose operation with real values increases bandwidth and processing requirements and leads to approximate solutions. Our paper is a major departure from current literature since the operation of each node relies on quantized communication. Utilization of quantized values allows more efficient usage of network resources, and leads to the calculation in finite time of the exact solution without any errors due to quantization. Therefore, our proposed distributed algorithm introduces a novel approach for data clustering with efficient (quantized) communication.

II. Mathematical Notation

The sets of real, rational, and integer numbers are denoted by $\mathbb{R}, \mathbb{Q}$, and $\mathbb{Z}$, respectively. The symbol $\mathbb{Z}_{\geq 0}$ ($\mathbb{Z}_{>0}$) denotes the set of nonnegative (positive) integer numbers. The set $\mathbb{Z}_{\leq 0}$ ($\mathbb{Z}_{<0}$) denotes the set of nonpositive (negative) integer numbers. For any real number $a \in \mathbb{R}$, $\lfloor a \rfloor$ denotes the greatest integer less than or equal to a (i.e., the floor of a), and $\lceil a \rceil$ denotes the least integer greater than or equal to a (i.e., the ceiling of a). Vectors are denoted by small letters, and matrices are denoted by capital letters. The transpose of matrix A is denoted by A^T. For matrix $A \in \mathbb{R}^{n \times n}$, $A_{(ij)}$ denotes the entry at row i and column j. For a vector $a \in \mathbb{R}^n$,

Apostolos I. Rikos and K. H. Johansson are with the Division of Decision and Control Systems, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden. E-mails: {rikos,kallej}@kth.se.

Gabriele Oliva is with the Unit of Automatic Control, Department of Engineering, Università Campus Bio-Medico di Roma, via Álvaro del Portillo 21, 00128, Rome, Italy. E-mail: g.oliva@unicampus.it.

C. N. Hadjicostis is with the Department of Electrical and Computer Engineering, University of Cyprus, 1678 Nicosia, Cyprus. E-mail: chadjic@ucy.ac.cy.

This work was supported by the Knut and Alice Wallenberg Foundation and the Swedish Research Council.

$a_{(i)}$ denotes the entry at position i. The all-ones vector is denoted as $\mathbf{1}$ and the identity matrix is denoted as I (of appropriate dimensions).

Graph Theoretic Notions. Let us consider a network of n nodes ($n > 2$) where each node can communicate only with its immediate neighbors. The communication topology is captured by a directed graph (digraph) defined as $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$. In digraph $\mathcal{G}_d$, $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes with cardinality $n = |\mathcal{V}| \geq 2$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} - \{(v_j, v_j) \mid v_j \in \mathcal{V}\}$ is the set of edges (self-edges excluded) with cardinality $m = |\mathcal{E}|$. A directed edge from node v_i to node v_j is denoted by $m_{ji} \triangleq (v_j, v_i) \in \mathcal{E}$, and captures the fact that node v_j can receive information from node v_i (but not the other way around). We assume that the given digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ is *strongly connected*. This means that for each pair of nodes $v_j, v_i \in \mathcal{V}$, $v_j \neq v_i$, there exists a directed *path*[1] from v_i to v_j. The diameter D of a digraph is the longest shortest path between any two nodes $v_j, v_i \in \mathcal{V}$.

The subset of nodes that can directly transmit information to node v_j is called the set of in-neighbors of v_j and is represented by $\mathcal{N}_j^- = \{v_i \in \mathcal{V} \mid (v_j, v_i) \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^-$ is called the *in-degree* of v_j and is denoted by $\mathcal{D}_j^- = |\mathcal{N}_j^-|$. The subset of nodes that can directly receive information from node v_j is called the set of out-neighbors of v_j and is represented by $\mathcal{N}_j^+ = \{v_l \in \mathcal{V} \mid (v_l, v_j) \in \mathcal{E}\}$. The cardinality of $\mathcal{N}_j^+$ is called the *out-degree* of v_j and is denoted by $\mathcal{D}_j^+ = |\mathcal{N}_j^+|$.

Remark 1. *Note that the communication topology (which is captured by a directed graph) denotes that nodes are able to wirelessly transmit/receive messages to their immediate neighbors (i.e., receive messages from their in-neighbors and transmit messages to their out-neighbors), thus exchanging their stored information in a distributed fashion.*

Node Operation. We assume that each node v_j can directly transmit messages to each out-neighbor; however, it cannot necessarily receive messages (at least not directly) from them. In the proposed distributed algorithm, each node v_j assigns a *unique order* in the set $\{0, 1, ..., \mathcal{D}_j^+ - 1\}$ to each of its outgoing edges m_{lj}, where $v_l \in \mathcal{N}_j^+$. More specifically, the order of link (v_l, v_j) for node v_j is denoted by P_{lj} (such that $\{P_{lj} \mid v_l \in \mathcal{N}_j^+\} = \{0, 1, ..., \mathcal{D}_j^+ - 1\}$). This unique predetermined order is used during the execution of the proposed algorithm as a way of allowing node v_j to transmit messages to its out-neighbors in a *round-robin*[2] fashion.

Distributed Max- and Min-Consensus. The distributed max-consensus algorithm calculates the maximum value of the network in a finite number of time steps [11]. The intuition of the algorithm is the following: if every node v_j in the network performs the following update

$$x_j[\mu + 1] = \max_{v_i \in \mathcal{N}_j^- \cup \{v_j\}} \{x_i[\mu]\}, \ \mu = 0, 1, ..., s, \quad (1)$$

then all nodes converge to the maximum value among the initial values $\{x_i[0] \mid v_i \in \mathcal{V}\}$ of all nodes in a finite number of steps s, where $s \leq D$ (see, e.g., [12, Theorem 5.4]). Note here that similar results hold also for the min-consensus algorithm, i.e., if the max operation is replaced by the min operation.

III. Problem Formulation

A. k-means Clustering

The problem we present in this paper is borrowed from [7], but is adjusted in the context of quantized communication over directed networks. Specifically, let us consider a set of n observations $x_1, \ldots, x_n$, where $x_i \in \mathbb{R}^d$ for $i \in \{1, 2, ..., n\}$. Each observation x_i is assigned to each node v_i, respectively. In the k-means clustering problem, we want to partition the n observations into k sets or *clusters* $\mathcal{C} = \{C_1, \ldots, C_k\}$ (where $k \leq n$) so we can minimize the sum of some measure of distance of the elements within every cluster. Specifically, we want to find a set of centroids $\mathbf{c}_1, \ldots, \mathbf{c}_k$ (where $\mathbf{c}_\gamma \in \mathbb{R}^d$, for $\gamma \in \{1, 2, ..., k\}$), each associated to a cluster, and a clustering assignment $r_{\gamma j}$ (where $r_{\gamma j} = 1$ means that node v_j belongs in cluster C_γ, $r_{\gamma j} = 0$ otherwise), which solves the following optimization problem:

$$\mathcal{D} = \underset{\{r_{\gamma j}\}, \{\mathbf{c}_\gamma\}}{\arg\min} \sum_{\gamma=1}^{k} \sum_{j=1}^{n} r_{\gamma j} ||x_j - \mathbf{c}_\gamma||^2, \quad (2)$$

$$\text{s.t.} \ \sum_{\gamma=1}^{k} r_{\gamma j} = 1, \ \text{for all } j = 1, 2, ..., n, \quad (3)$$

$$r_{\gamma j} \in \{0, 1\}, \ \text{for } \gamma = 1, 2, ..., k, \ j = 1, 2, ..., n. \quad (4)$$

The problem in (2)–(4) is hard to solve exactly when n and k are large,[3] thus calling for approximate solutions. In particular, the k-means algorithm represents a successful strategy to compute a locally optimal solution to the above problem. The intuition of the k-means algorithm is that it starts with a random set of k centroids $\mathbf{c}_1(1), \ldots, \mathbf{c}_k(1)$, and alternates at each step T between an *assignment* and a *refinement* phase.

Assignment phase: Each observation x_λ is assigned to the set characterized by the nearest centroid, i.e.:

$$C_\gamma(T) = \\ = \{x_\lambda : ||x_\lambda - \mathbf{c}_\gamma(T)||^2 \leq ||x_\lambda - \mathbf{c}_{\gamma'}(T)||^2, \ \gamma, \gamma' \in [1, k]\}. \quad (5)$$

Refinement phase: Each centroid $\mathbf{c}_i(T+1)$ is updated as:

$$\mathbf{c}_\gamma(T + 1) = \frac{\sum_{v_j \in C_\gamma(T)} x_j}{|C_\gamma(T)|} \quad (6)$$

[1] A directed *path* from v_i to v_j exists if we can find a sequence of nodes $v_i \equiv v_{l_0}, v_{l_1}, \ldots, v_{l_t} \equiv v_j$ such that $(v_{l_{\tau+1}}, v_{l_\tau}) \in \mathcal{E}$ for $\tau = 0, 1, \ldots, t-1$.

[2] When executing the deterministic protocol, each node v_j transmits to its out-neighbors, one at a time, by following the predetermined order. The next time it transmits to an out-neighbor, it continues from the outgoing edge it stopped the previous time and cycles through the edges in a round-robin fashion.

[3] The problem is NP-hard in general Euclidean space $\mathbb{R}^d$, even for 2 clusters [13] and for a general number of clusters k, even in the plane [14].

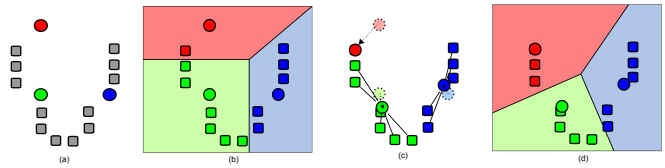

Fig. 1. Example of execution of k-means algorithm (source: Wikipedia Commons available under GNU Free Documentation License v. 1.2).

The two steps are iterated until convergence (i.e., if the centroids no longer change) or up to a maximum of M iterations.

Fig. 1 is an example of an execution of the algorithm for a set of $n = 12$ observations in $\mathbb{R}^2$ and for $k = 3$. In this case, the observations coincide with the agent's positions (i.e., the agents are clustered according to their position). Fig. 1 (a) shows with circles the initial centroids. Fig. 1 (b) and Fig. 1 (c) report the assignment and refinement phases for the first step. Fig. 1 (d) depicts the assignment phase for the second step.

The k-means algorithm is known to converge to a local optimum value, while there is no guarantee to converge to the global optimum [15]. However, given the complexity of the problem at hand, the k-means algorithm is de facto the most diffused heuristic algorithm: indeed "ease of implementation, simplicity, efficiency, and empirical success are the main reasons for its popularity" [16]. Furthermore, note that the convergence of the algorithm strongly depends on the initial choice of the centroids. Therefore, a common practice is to execute the algorithm several times – each time with different initial conditions – and select the best solution.

B. Modification of k-means Clustering Problem: Finite-Time k-means Clustering with Quantized Communication

In this paper, we develop a distributed algorithm that allows nodes to find a locally optimal solution to the problem **P1** presented below, while transmitting quantized information via available communication links. We assume that the transmitted quantized messages are integer-valued (this comprises a class of quantization effects such as uniform quantization).

P1. Consider a static strongly connected digraph $\mathcal{G}_d$, where each node v_j is endowed with a quantized state $x_j \in \mathbb{R}^d$. We aim at developing an algorithm which calculates in a distributed fashion a set of centroids $\mathbf{c}_1, \ldots, \mathbf{c}_k$ (where $\mathbf{c}_\gamma \in \mathbb{R}^d$, for $\gamma \in \{1, 2, ..., k\}$) and association variables $r_{\gamma j}$, which represent a locally optimal solution to the optimization problem presented in (2)–(4). The proposed algorithm converges in a finite number of time steps, upper bounded by a polynomial function which depends on the communication network. During the proposed algorithm each node transmits quantized information and ceases transmissions once convergence has been achieved.

IV. Finite-Time k-means Clustering with Quantized Communication

In this section we propose a distributed algorithm which solves problem **P1** in Section III-B. For developing our proposed algorithm in this paper, we first present an extended version of the algorithm in [17] which is important for our subsequent development. Then, we present the distributed algorithm which solves Problem **P1**.

A. Multidimensional Deterministic Exact Quantized Average Consensus

In this section, we present an extended version of the deterministic algorithm in [17]. In this version, each node is able to calculate the exact average of the initial states in a deterministic fashion after a finite number of time steps for the case where the state of each node is an integer vector (i.e., $y_j[\mu] \in \mathbb{Z}^d$, where $\mu, d \in \mathbb{Z}_{>0}$). The proposed algorithm is detailed as Algorithm 1 below.

The intuition of Algorithm 1 is the following. Each node v_j receives the mass variables $y_i[k]$ and $z_i[k]$ from its in-neighbors $v_i \in \mathcal{N}_j^-$ and sums them along with its stored mass variables ($y_j[k]$ and $z_j[k]$). During the event-triggered conditions C1–C5, node v_j compares each element of the received vectors against its stored vectors. According to the event-triggered conditions, it decides whether it will update its state variables and/or perform transmission towards one of its out-neighbors. If it performs a transmission, it sets its stored mass variables equal to zero and repeats the procedure.

Definition 1. *The system is able to achieve exact quantized average consensus in the form of a quantized fraction if, for every $v_j \in \mathcal{V}$, there exists $\mu_0 \in \mathbb{Z}_+$ so that for every $v_j \in \mathcal{V}$ we have*

$$q_j^s[\mu] = q, \tag{7}$$

for $\mu \geq \mu_0$, where q is the real average of the initial states defined as

$$q = \frac{1}{n}\sum_{l=1}^{n} y_l[0]. \tag{8}$$

Let us now consider the following setup.

Setup 1: Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$ has an initial quantized state $y_j[0] \in \mathbb{Z}^d$. During the execution of the Algorithm 1, at time step μ_1, there is at least one node $v_{j'} \in \mathcal{V}$, for which

$$z_{j'}[\mu_1] \geq z_i[\mu_1], \ \forall v_i \in \mathcal{V}. \tag{9}$$

Then, among the nodes $v_{j'}$ for which (9) holds, there is at least one node v_j, for which there exist some maximal dimension $\dim_j$, such that

$$y_{j(k')}[\mu_1] \geq y_{j'(k')}[\mu_1], \ v_j, v_{j'} \in \{v_i \in \mathcal{V} \mid (9) \text{ holds}\}, \tag{10}$$

for $k' \in \{1, 2, ..., \dim_j\}$. For notational convenience we will call the mass variables of node v_j (or the nodes, if they are

Algorithm 1 Multidimensional Deterministic Exact Quantized Average Consensus Algorithm

Input: A strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$ has an initial quantized state $y_j[1] \in \mathbb{Z}^d$.

Initialization: Every node $v_j \in \mathcal{V}$ does the following:

- assigns to each of its outgoing edges $\{(v_l, v_j) \mid v_l \in \mathcal{N}_j^+\}$ a *unique order* P_{lj} in the set $\{0, 1, ..., \mathcal{D}_j^+ - 1\}$;
- sets $tr^{(j)} = 0$ and $e = tr^{(j)}$;
- sets $z_j[1] = 1$, $z_j^s[1] = z_j[1]$ and $y_j^s[1] = y_j[1]$ (which means that $q_j^s[1] = y_j^s[1]/z_j^s[1]$);
- chooses out-neighbor $v_l \in \mathcal{N}_j^+$ according to the predetermined order P_{lj} (initially, it chooses $v_l \in \mathcal{N}_j^+$ such that $P_{lj} = 0$) and transmits $z_j[1]$ and $y_j[1]$ to this out-neighbor. Then, it sets $y_j[1] = 0$, $z_j[1] = 0$, $\text{pass}_j = 0$;
- sets $tr^{(j)} = tr^{(j)} + 1$ and $e = tr^{(j)} \mod \mathcal{D}_j^+$;

Iteration: For $\mu = 1, 2, \ldots$, each node $v_j \in \mathcal{V}$, does the following:

- receives $y_i[\mu]$ and $z_i[\mu]$ from its in-neighbors $v_i \in \mathcal{N}_j^-$ and sets

$$y_j[\mu+1] = y_j[\mu] + \sum_{v_i \in \mathcal{N}_j^-} w_{ji}[\mu] y_i[\mu],$$

and

$$z_j[\mu+1] = z_j[\mu] + \sum_{v_i \in \mathcal{N}_j^-} w_{ji}[\mu] z_i[\mu],$$

where $w_{ji}[\mu] = 0$ if no message is received (otherwise $w_{ji}[\mu] = 1$);
- sets $\text{pass}_j = 1$, sets $\text{dim}_j = 1$;
- Event Trigger Conditions: **while** $\text{dim}_j \leq d$ **then**
  C1: **if** $z_j[\mu+1] > z_j^s[\mu]$ **break**;
  C2: **if** $z_j[\mu+1] < z_j^s[\mu]$ **sets** $\text{pass}_j = 0$, **break**;
  C3: **if** $z_j[\mu+1] = z_j^s[\mu]$ and $y_{j(\text{dim}_j)}[\mu+1] > y^s_{j(\text{dim}_j)}[\mu]$ **break**;
  C4: **if** $z_j[\mu+1] = z_j^s[\mu]$ and $y_{j(\text{dim}_j)}[\mu+1] = y^s_{j(\text{dim}_j)}[\mu]$ **sets** $\text{dim}_j = \text{dim}_j + 1$;
  C5: **if** $z_j[\mu+1] = z_j^s[\mu]$ and $y_{j(\text{dim}_j)}[\mu+1] < y^s_{j(\text{dim}_j)}[\mu]$ **sets** $\text{pass}_j = 0$, **break**;
- **if** $\text{pass}_j = 1$:
  - sets $z_j^s[\mu+1] = z_j[\mu+1]$, $y_j^s[\mu+1] = y_j[\mu+1]$,

$$q_j^s[\mu+1] = \frac{y_j^s[\mu+1]}{z_j^s[\mu+1]}.$$

  - transmits $z_j[\mu+1]$ and $y_j[\mu+1]$ towards out-neighbor $v_\lambda \in \mathcal{N}_j^+$ for which $P_{\lambda j} = e$ and it sets $y_j[\mu+1] = 0$ and $z_j[\mu+1] = 0$. Then, it sets $tr^{(j)} = tr^{(j)} + 1$ and $e = tr^{(j)} \mod \mathcal{D}_j^+$.

Output: (7) holds for every $v_j \in \mathcal{V}$.

more than one) for which (9) and (10) hold as the "leading mass" (or "leading masses").

In the following theorem we present the deterministic convergence of Algorithm 1. The proof of the theorem is similar to Proposition 1 in [17] and is omitted.

Theorem 1. *Under Setup* 1 *we have that the execution of Algorithm 1 allows each node $v_j \in \mathcal{V}$ to reach quantized average consensus after a finite number of steps $\mathcal{S}_t$, bounded by $\mathcal{S}_t \leq nm^2$.*

B. Finite-Time k-means Clustering Algorithm with Quantized Communication

We now present a distributed algorithm which solves Problem **P1** presented in Section III-B. The proposed algorithm is detailed as Algorithm 2 below.

Assumption 1. *Every node $v_j \in \mathcal{V}$ knows the diameter of the network D (or an upper bound D').*

Assumption 2. *Each node v_j knows the initial set of centroids $C[0] = [\boldsymbol{c}_1[0], \boldsymbol{c}_2[0], ..., \boldsymbol{c}_k[0]] \in \mathbb{R}^{d \times k}$ ($k < n$).*

Assumption 1 is a necessary condition for the min- and max-consensus algorithm, so that each node v_j is able to determine whether convergence has been achieved and thus our proposed algorithm can terminate. Note, however, that this assumption can be relaxed if we utilize the distributed algorithm in [18] instead of Algorithm 1. The algorithm in [18] converges to the exact real average in finite time without requiring knowledge of the network diameter D. Assumption 2 is a necessary condition so that each node can calculate the updated value of the centroids without having to communicate their real values to other nodes (because communication is restricted to quantized values).

Remark 2. *Regarding Assumption 2, previous work in [7] ensures that one node is elected as the leader node and propagates the real values of the centroids to every node. Note that in our case, nodes communicate by exchanging quantized values. Therefore, each node needs to know the initial set of centroids in order to calculate their updated values without the need of a leader node (which transmits the updated value of the centroids to every node in the network). When the initial set of centroids is known only to a certain leader node, then the leader node can propagate the set of centroids to every node if the initial set of centroids comprises quantized values. Thus, after D time steps, every node in the network will know the initial centroids and Assumption 2 will be fulfilled.*

We now describe the main operations of Algorithm 2. The initialization involves the following steps:

Initialization. Centroid Selection and Unique Order: Each node $v_j \in \mathcal{V}$ has a quantized state $x_j \in \mathbb{Z}^d$. Then, it assigns to each of its outgoing edges a *unique order*.

The iteration involves the following steps:

Iteration - Step 1. Cluster Assignment and Labeled Multidimensional Deterministic Exact Quantized Average Consensus: At each step μ, each node v_j assigns its

value x_j to the nearest centroid according to (5) (since each node knows the set of initial centroids $C(0) = [\mathbf{c}_1(0), \mathbf{c}_2(0), ..., \mathbf{c}_k(0)] \in \mathbb{R}^{d\times k}$ ($k < n$)). This means that it sets $r_{\lambda j} = 1$ (where $\mathbf{c}_\lambda(0)$ is the nearest centroid) with respect to (3). Then, it performs k quantized average consensus operations – each operation is done with the states of the nodes that belong in the same cluster. Specifically, each node v_j executes k times (in parallel) Algorithm 1 in Section IV-A. Each execution is done with the nodes $\{v_i \in \mathcal{V} \mid r_{\lambda i} = r_{\lambda j} = 1\}$ (i.e., the nodes that belong in the same cluster). In this way, each node calculates the *exact* updated value of every centroid in finite time.

Iteration - Step 2. Labeled Distributed Stopping: Every node v_j performs k parallel $\min-$ and $\max-$consensus operations every D time steps as described in Section II. Each operation is done with the states of the nodes that belong in the same cluster (i.e., with nodes v_i, v_j for which $r_{\lambda j} = r_{\lambda i} = 1$). In this way, each node is able to determine whether convergence has been achieved and the updated set of centroids $\mathbf{c}_\gamma[T+1]$, for every $\gamma \in \{1, 2, ..., k\}$, has been calculated.

Iteration - Step 3. Centroid Update, Cluster Assignment and Algorithm Termination: Once all k executions of Algorithm 1 have converged, each node v_j updates the stored set of centroids. Then, it assigns its value x_j to the nearest updated centroid according to (5). It checks if the previous centroid values are equal to the new centroid values. If this condition holds for each node v_j, the operation of the algorithm is terminated. Otherwise, the iteration is repeated.

The flowchart for the operation of each node during Algorithm 2 is shown in Fig. 2. In the flowchart, we can see that initially each node assigns a unique order to its outgoing edges and it also assigns its value to the cluster characterized by the nearest centroid. Then, for each of the k clusters it executes (i) "Labeled Multidimensional Deterministic Exact Quantized Average Consensus" (shown in Algorithm 1 for the case where we have one cluster $k = 1$), and (ii) "Labeled Distributed Stopping". This allows the calculation of the new centroid values. Then, it checks if the previous centroid values are equal to the new centroid values. If this condition holds, the algorithm has converged and every node terminates its operation. Otherwise, the process is repeated.

Next, we show that, during the operation of Algorithm 2, each node v_j is able to (i) calculate a set of centroids $\mathbf{c}_1, \ldots, \mathbf{c}_k$ that fulfill (2) after a finite number of time steps, and (ii) terminate its operation once convergence has been achieved. Due to space limitations, we omit the proof of the theorem below; it will be available in an extended version of our paper.

Theorem 2. *Consider a strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$ has an initial quantized state $x_j \in \mathbb{Z}^d$, and knows the k initial clusters (where $k < n$) and their initial centroids $C[0] = [\boldsymbol{c}_1[0], \boldsymbol{c}_2[0], ..., \boldsymbol{c}_k[0]] \in \mathbb{R}^{d\times k}$. During the operation of Algorithm 2, each node v_j is able to address problem **P1** in Section III-B after a finite number of time steps C_t bounded by*

$$C_t \leq T(D + nm^2), \tag{14}$$

where T is the number of new centroid calculations (see [19]) until (11) holds, and D is the diameter of network $\mathcal{G}_d$.

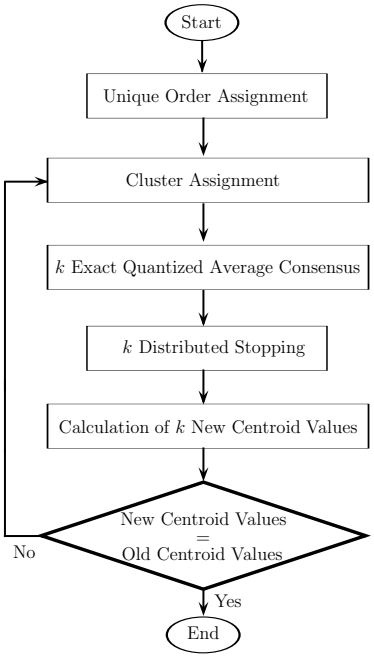

Fig. 2. Flowchart for operation of each node during Algorithm 2.

Remark 3. *In Theorem 2, the number of new centroid calculations T until (11) holds is finite [19]. More specifically, T depends on the k^n ways to partition n observations into k clusters. Since k^n is finite number, we have that T is also a finite number. As a result, the number of new centroid calculations T is finite, and Algorithm 2 converges in finite time (see Theorem 2). As mentioned earlier, the maximum number of new centroid calculations is sometimes fixed to some integer M.*

V. Simulation Results

We now present simulation results to illustrate the behavior of Algorithm 2 over several examples.

Evaluation over a Random Network of 100 Nodes. We execute Algorithm 2 over a random digraph of 100 nodes with diameter $D = 4$. The 100 nodes are randomly placed in a $[50, 50] \times [50, 50]$ region with uniform probability. During the operation of Algorithm 2 we partition nodes into $k = 3$ clusters and calculate the centroid values that fulfill (2) in finite time. In this case, the observations coincide with the agent's positions (i.e., the agents are clustered according to their position). In Fig. 3 (A) the initial positions of the 3 centroids are marked by red, blue and green crosses, the nodes are marked with circles, and each circle color is the color of the nearest initial centroid (i.e., nodes are marked red, blue, or green color). In Fig. 3 (B) we show the trajectories of the centroids which are marked with red, blue, or green lines according to the color of the centroid. We can see that after $T = 10$ the centroid values fulfill

Algorithm 2 Finite-Time Quantized k-means Algorithm

Input: A strongly connected digraph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Each node $v_j \in \mathcal{V}$ has an initial quantized state $x_j \in \mathbb{Z}^d$, and has knowledge of D. Each node v_j knows the number of clusters $k < n$ and the initial centroids $C[0] = [\mathbf{c}_1[0], \mathbf{c}_2[0], ..., \mathbf{c}_k[0]] \in \mathbb{R}^{d\times k}$.

Initialization: Each node v_j sets $\text{flag}_j = 0$ and assigns to each of its outgoing edges $v_l \in \mathcal{N}_j^+$ a *unique order* P_{lj} in the set $\{0, 1, ..., \mathcal{D}_j^+ - 1\}$.

Iteration: For $\mu = 1, 2, \ldots$, each node $v_j \in \mathcal{V}$, does the following:

- **while** $\text{flag}_j = 0$ **then**
  - sets $r_{\lambda j} = 1$, where λ is such that $||x_j - c_\lambda[\mu]|| \leq ||x_j - c_\gamma[\mu]||$, and $\gamma \in \{1, 2, ..., k\} \setminus \{\lambda\}$;
  - sets $x_j^{cl}[\mu] = x_j$, for $cl = \lambda$, where $r_{\lambda j} = 1$, and $x_j^{cl}[\mu] = 0$, for $cl = \{1, 2, ..., k\} \setminus \{\lambda\}$;
  - calls Algorithm 2.A;
- **if**

$$\mathbf{c}_\gamma(T+1) = \mathbf{c}_\gamma(T), \text{ for every } \gamma \in \{1, 2, ..., k\}, \quad (11)$$

then $\text{flag}_j = 1$;

Output: (2), (3), (4) hold for every $v_j \in \mathcal{V}$.

(11). This means that the nodes are able to determine that convergence has been achieved and thus proceed to terminate their operation. In Fig. 4 we show the evolution of the Distance Objective Function $F(T)$ defined as

$$F(T) = \sum_{\gamma=1}^{k} \sum_{v_j \in C_\gamma(T)} ||x_j - \mathbf{c}_\gamma(T)||^2, \quad (15)$$

for the network of Fig. 3. We can see that $F[T]$ is non-increasing over time. Furthermore, we can see that $F[9] = F[10]$. This means that for $T = 10$ the centroid values fulfill (11) and nodes terminate their operation (see Fig. 3 (B)).

Evaluation over 1000 Random Networks of 1000 Nodes. We execute of Algorithm 2 over 1000 random networks each consisting of 1000 nodes, with diameter $D \in \{3, 4, 5\}$. During the operation of Algorithm 2, we aim to partition nodes into $k = 3$ clusters. For each of the 1000 simulations, the nodes and the centroid positions are randomly placed in the region $[100, 100] \times [100, 100]$ with uniform probability. We present the distribution $\mathcal{F}[T]$ of the new centroid calculations T until (11) holds for 100 simulations of Algorithm 2. Furthermore, we present the average value $\overline{F}[T]$ of the Distance Objective Function $F[T]$ in (15), averaged over the 1000 simulations of Algorithm 2.

In Fig. 5 (A), for 1000 executions of Algorithm 2 we have that $\overline{F}[T]$ almost converges after 17 centroid calculations T. Also, note that in Fig. 5 (A), $\overline{F}[T]$ is plotted for $T \in \{1, 2, ..., 56\}$, where 56 is the maximum value of T for Algorithm 2 to converge over the 1000 executions. In Fig. 5 (B), we have that the average value of T for Algorithm 2 to converge over the 1000 executions is 17.39. The minimum value of T is 5, and the maximum is 56 (also seen in Fig. 5

Algorithm 2.A Extended Labeled Multidimensional Deterministic Quantized Average Consensus with Labeled Distributed Stopping

Input: D, T, μ, $x_j^{cl}[\mu]$ for $cl = \{1, 2, ..., k\}$, P_{lj} for every $v_l \in \mathcal{N}_j^+$;

Initialization: $y_j^{cl}[1] = x_j^{cl}[\mu]$ for $cl = \{1, 2, ..., k\}$;

Iteration: For $\mu' = 1, 2, \ldots$, each node $v_j \in \mathcal{V}$, does the following:

- **while** $\text{flag}_j = 0$ **then**
  - **if** $\mu' \mod D = 1$ **then** sets $M_j^{cl} = x_j^{cl}[\mu']/z_j^{cl}[\mu']$, $m_j^{cl} = x_j^{cl}[\mu']/z_j^{cl}[\mu']$, where $z_j[\mu'] = 1$, $cl = \{1, 2, ..., k\}$;
  - broadcasts M_j^{cl}, m_j^{cl}, $cl = \{1, 2, ..., k\}$, to every $v_j \in \mathcal{N}_j^+$;
  - receives M_i^{cl}, m_i^{cl} from every $v_i \in \mathcal{N}_j^-$, $cl = \{1, 2, ..., k\}$;
  - sets $M_j^{cl} = \max_{v_i \in \mathcal{N}_j^- \cup \{v_j\}} M_i^{cl}$, and $m_j^{cl} = \min_{v_i \in \mathcal{N}_j^- \cup \{v_j\}} m_j^{cl}$;
  - executes Iteration Steps of Algorithm 1 for each $cl = \{1, 2, ..., k\}$ with initial state $y_j^{cl}[\mu]$;
  - receives $z_i^{cl}[\mu']$, $y_i^{cl}[\mu']$ from $v_i \in \mathcal{N}_j^-$ and sets

$$y_j^{cl}[\mu'+1] = y_j^{cl}[\mu'] + \sum_{v_i \in \mathcal{N}_j^-} w_{ji}[\mu']\, y_i^{cl}[\mu'], \quad (12)$$

$$z_j^{cl}[\mu'+1] = z_j^{cl}[\mu'] + \sum_{v_i \in \mathcal{N}_j^-} w_{ji}[\mu']\, z_i^{cl}[\mu'], \quad (13)$$

    where $w_{ji}[\mu'] = 1$ if node v_j receives a message from $v_i \in \mathcal{N}_j^-$ at iteration μ' (otherwise, $w_{ji}[\mu'] = 0$);
  - **if** $\mu' \mod D = 0$ **then**, **if** $M_j^{cl} = m_j^{cl}$, for every $cl = \{1, 2, ..., k\}$ **then** sets $\mathbf{c}_{cl}[T+1] = q_j^{s,cl}[\mu']$ for every $cl = \{1, 2, ..., k\}$ and sets $\text{flag}_j = 1$.

Output: $\mathbf{c}_{cl}[T+1]$ for every $cl = \{1, 2, ..., k\}$.

(A)). Furthermore, we can see that in most cases, the required T for Algorithm 2 to converge over the 1000 executions is in the set $T \in \{8, 9, ..., 20\}$.

VI. Conclusions and Future Directions

In this paper, we have considered the problem of k-means clustering over a directed network. We presented a novel algorithm which is able to address the k-means clustering problem in a fully distributed fashion. We showed that our algorithm converges after a finite number of time steps, and we provided a deterministic upper bound on convergence time which relies on the network parameters. Finally, we demonstrated the operation of our proposed algorithm and compared its performance against other algorithms in the existing literature.

References

[1] Y. Li, J. Qi, X. Chu, and W. Mu, "Customer segmentation using k-means clustering and the hybrid particle swarm optimization algorithm," *The Computer Journal*, 2022.

[2] M. Jain, G. Kaur, and V. Saxena, "A k-means clustering and SVM based hybrid concept drift detection technique for network anomaly detection," *Expert Systems with Applications*, p. 116510, 2022.

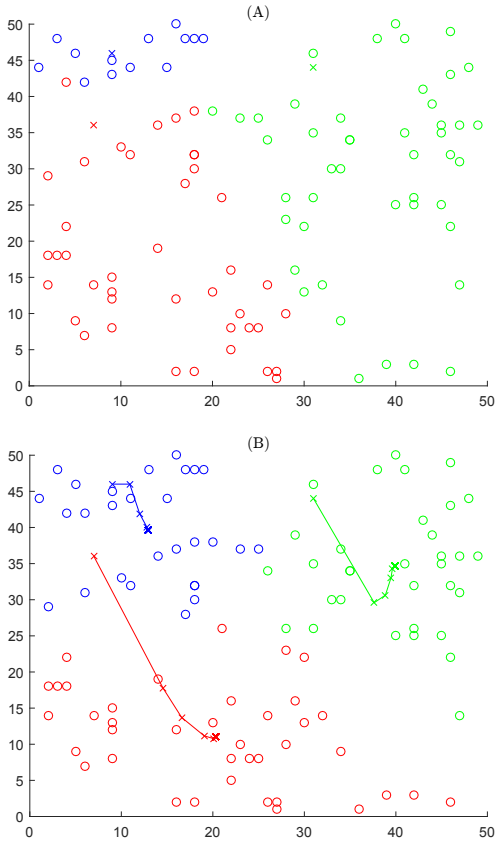

Fig. 3. Execution of Algorithm 2 over a random directed network with 100 nodes and 3 clusters. Positions of nodes are marked with circles and centroid positions are marked with red, blue and green crosses. (A) Initial centroid positions, and initial position and cluster assignment for every node. (B) Centroid trajectories, final centroid positions, and final cluster assignment for every node.

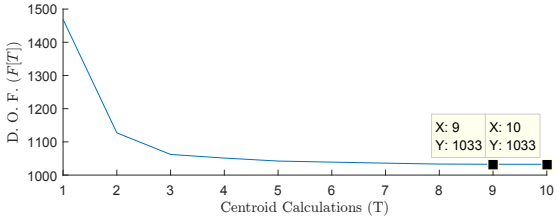

Fig. 4. Evolution of Distance Objective Function $F[T]$ during execution of Algorithm 2 over a random directed network with 100 nodes and 3 clusters.

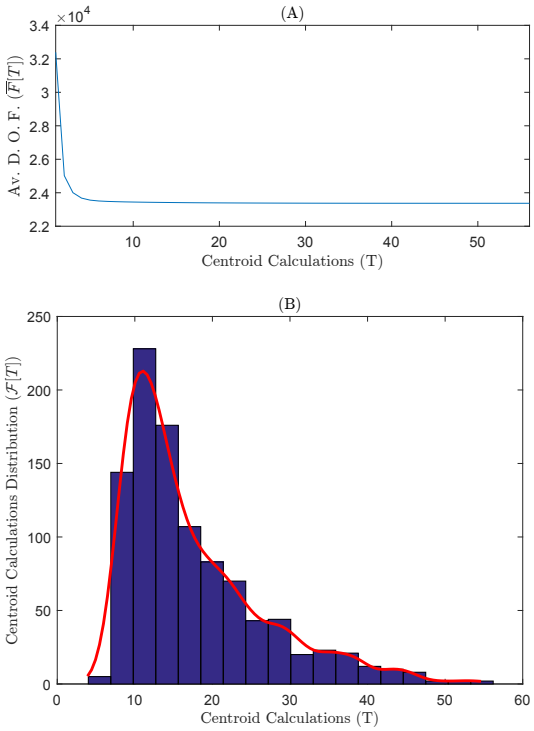

Fig. 5. Executions of Algorithm 2 over 1000 random directed networks of 1000 nodes and $k = 3$ clusters. (A) Evolution of Average Distance Objective Function $\overline{F}[T]$ during execution of Algorithm 2, averaged over 1000 executions. (B) Distribution $\mathcal{F}[T]$ of number of new centroid calculations T until (11) holds, for 1000 executions.

[3] S. Ferjaoui, "Data: The new form of wealth and power," *IEEE Potentials*, vol. 39, no. 6, pp. 6–10, 2020.

[4] G. Oliva, A. Gasparri, A. Fagiolini, and C. N. Hadjicostis, "Distributed computing over encrypted data," in *IEEE Conference on Decision and Control*, 2017, pp. 1584–1589.

[5] I. S. Dhillon and D. S. Modha, "A data-clustering algorithm on distributed memory multiprocessors," in *Workshop on Large-Scale Parallel KDD Systems, SIGKDD*, 2000, pp. 245–260.

[6] S. Bandyopadhyay, C. Giannella, U. Maulik, H. Kargupta, K. Liu, and S. Datta, "Clustering distributed data streams in peer-to-peer environments," *Information Sciences*, vol. 176, no. 14, pp. 1952–1985, 2006.

[7] G. Oliva, R. Setola, and C. N. Hadjicostis, "Distributed k-means algorithm," *arXiv preprint arXiv:1312.4176*, 2015.

[8] J. Qin, W. Fu, H. Gao, and W. X. Zheng, "Distributed k-means algorithm and fuzzy c-means algorithm for sensor networks based on multiagent consensus theory," *IEEE Transactions on Cybernetics*, vol. 47, no. 3, pp. 772–783, 2017.

[9] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based k-means algorithm for distributed learning using wireless sensor networks," in *Proceedings of the Workshop on Sensors, Signal, Info. Process.*, 2008, pp. 11–14.

[10] L. Faramondi, G. Oliva, R. Setola, and C. N. Hadjicostis, "Distributed c-means clustering via broadcast-only token passing," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 1, pp. 315–325, 2019.

[11] J. Cortés, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, pp. 726–737, March 2008.

[12] S. Giannini, D. Di Paola, A. Petitti, and A. Rizzo, "On the convergence of the max-consensus protocol with asynchronous updates," in *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2013, pp. 2605–2610.

[13] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hardness of Euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.

[14] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, "The planar k-means problem is NP–hard," *WALCOM: Algorithms and Computation*, pp. 274–285, 2009.

[15] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *in Proceedings of the* 5^{th} *Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, 1967.

[16] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[17] A. I. Rikos and C. N. Hadjicostis, "Event-triggered quantized average consensus via ratios of accumulated values," *IEEE Transactions on Automatic Control*, vol. 66, no. 3, pp. 1293–1300, 2020.

[18] A. I. Rikos, C. N. Hadjicostis, and K. H. Johansson, "Finite time exact quantized average consensus with limited resources and transmission stopping for energy-aware networks," *arXiv preprint arXiv:2110.00359*, 2021.

[19] S. Z. Selim and M. A. Ismail, "K-means-type algorithms: A generalized convergence theorem and characterization of local optimality," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 1, pp. 81–87, 1984.