Virtual Clustering: A Communication Cost Reduction Strategy for Distributed Consensus-Based Estimation in Cooperative Networks

Shengdi Wang, Guang Xu, Henning Paul, and Armin Dekorsy
Department of Communications Engineering, University of Bremen, Bremen, Germany
Email: {wang, xu, paul, dekorsy}@ant.uni-bremen.de

***Abstract*—In this paper, we consider the problem of distributed consensus-based estimation in cooperative networks, e.g., wireless sensor networks (WSNs). To solve this problem and achieve an accurate consensus-based estimate solution, many iterative distributed algorithms require the information exchange among nodes at each iteration suffering from huge communication overhead. In our previous work, a new strategy of virtual clustering was discussed with the purpose of reducing communication overhead for distributed consensus-based estimation. By classifying the data using virtual clusters, data with reduced size will be transmitted during the distributed processing. Here, we further propose two methods to reduce the size of transmitted data for arbitrary network topologies. One method is based on finding the shortest path in a network and the other relies on linear independence of constraint qualification (LICQ). The study shows that both methods can successfully reduce communication overhead. Moreover, the second method outperforms the first one and provides the optimal communication cost for the distributed consensus-based estimation.**

I. INTRODUCTION

Cooperative communication over networks and its applications have been widely discussed by the research community in recent years, e.g., monitoring and detecting environmental parameters or spacial fields via wireless sensor networks (WSNs) [1]-[3], tracking of objects or exploration of an area using multi-agent systems (MASs) [4]-[6] as well as multi-user detection in dense small-cell (SC) mobile networks [7]. In a common scenario, a group of nodes receives messages or senses physical parameters from sources with an objective to recover the common source information in a cooperative way. Thus, the local available information can be jointly used for successful reconstruction. One possible way to realize jointly reconstruction is performing in a centralized fashion, where a central node in the network is deployed as a 'fusion center' [8] to aggregate information of the whole network and performs centralized processing. However, the risk of a single point of failure and complicate routing protocol for multi-hops transmission in a large-scale system will restrict the application of centralized scheme. An alternative scheme for jointly reconstruction is processing in a distributed fashion, in which nodes are homogeneous with both processing and communication capabilities. Each node will perform the local estimation and only exchange information with its neighbors via inter-node links, which provides more flexible and robust estimation over the whole network compared to the centralized processing.

Consensus in distributed estimation is also an interesting topic investigated in a plenty of works. Among them, the average consensus scheme [9], [10] is extensively used. In this work, we consider distortions on the observations due to, e.g., fading channel, the common source messages cannot be properly recovered at each node using averaging operation. Thus, distributed consensus-based estimation by introducing consensus constraints to a general objective function of the estimation problem is preferred. This triggers the development of some primal and dual method-based algorithms [11]-[14], which are mainly focused on in this paper.

To obtain an accurate consensus-based estimation result, primal and dual method-based distributed algorithms involved in iterative processing suffer from huge communication cost, because they require the information exchange among nodes at each iteration to achieve the overall estimate solution. Considering efficiency and energy saving, it is necessary to reduce the communication overhead, and three aspects can be taken into account. Firstly, the acceleration of convergence speed of iterative algorithm can be pursued, e.g., [15]. Secondly, cutting down some transmitted variables can be considered, e.g., [14]. Thirdly, reducing the size or dimension of transmitted elements is also a feasible aspect such as the approaches described in [16] and [17]. Inspired by the works related to the third aspect, the virtual clustering (VC) strategy proposed in [18] is an effective communication cost reduction method for the distributed consensus-based estimation. Here, we specify the VC strategy by exploiting two methods to significantly reduce the amount of transmitted data for arbitrary network topologies.

The organization of this paper is summarized as follows. In Section II, the general estimation problem will be introduced. Then, we review the VC strategy of our previous work applied in distributed consensus-based estimation in Section III. Next, two methods to determine virtual data selection (VDS) matrix for partial transmission as the major part of this work will be discussed in Section IV. In Section V, numerical simulation results are given to show the advantages of our novel VC strategy based algorithm applying both methods. Finally, the work is summarized in Section VI.

II. Problem Statement

We consider a network with a set of nodes $\mathcal{J} := \{1, \cdots, J\}$ connected with assumed error-free links. These nodes forming a certain time-invariant topology can be represented by a geometric graph $\mathcal{G} = \{\mathcal{J}, \mathcal{E}\}$. Here, $\mathcal{E}$ describes the set of edges. We assume the graph is connected which means each node is able to reach any other node by multi-hops. The task of these nodes in the network is to observe data sent by U sources and estimate the overall source messages. In this scenario, we model each source and node to be equipped with N_{T} transmit and N_{R} receive antennas, respectively. Therefore, a total $N_{\mathrm{I}} = N_{\mathrm{T}} \cdot U$ system input from a stacked source vector $\boldsymbol{x} = [\boldsymbol{x}_1^{\mathrm{T}}, .., \boldsymbol{x}_U^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{C}^{N_{\mathrm{I}}}$ is monitored by each node $j \in \mathcal{J}$ resulting in the local observation $\mathbf{y}_j \in \mathbb{C}^{N_{\mathrm{R}}}$ determined by

$$\mathbf{y}_j = \mathbf{H}_j \boldsymbol{x} + \mathbf{n}_j \tag{1}$$

with the channel coefficient matrix $\mathbf{H}_j \in \mathbb{C}^{N_{\mathrm{R}} \times N_{\mathrm{I}}}$ known locally and zero mean additive white Gaussian noise $\mathbf{n}_j \in \mathbb{C}^{N_{\mathrm{R}}}$. In this work, we assume $N_{\mathrm{I}} > N_{\mathrm{R}}$, hence (1) is an underdetermined system. Each node can not estimate the source message $\boldsymbol{x}$ properly only use the local information. Therefore, jointly estimation is preferred to obtain an accurate and unique estimate solution for all nodes based on the global knowledge of $\mathbf{H}_j$ and $\mathbf{y}_j$. Here, the centralized Least-Square (LS) criterion [19] is adopted to minimize the overall sum of residuals:

$$\mathbf{x}_{\text{cen}} = \underset{\{\mathbf{x}' \in \mathbb{C}^{N_{\mathrm{I}}}\}}{\arg\min} \ \|\mathbf{y}_{\text{cen}} - \mathbf{H}_{\text{cen}} \mathbf{x}'\|^2, \tag{2}$$

where $\mathbf{y}_{\text{cen}} = \left[\mathbf{y}_1^{\mathrm{T}}, .., \mathbf{y}_J^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{C}^{J N_{\mathrm{R}}}$ and $\mathbf{H}_{\text{cen}} = \left[\mathbf{H}_1^{\mathrm{T}}, ..., \mathbf{H}_J^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{C}^{J N_{\mathrm{R}} \times N_{\mathrm{I}}}$ ($J N_{\mathrm{R}} > N_{\mathrm{I}}$) are stacked observation vector and stacked channel coefficient matrix, respectively. The problem in (2) can be solved in a central node by centralized processing. The solution of (2) can be obtained by, e.g., the Zero Forcing (ZF) approach resulting in

$$\mathbf{x}_{\text{ZF}} = (\mathbf{H}_{\text{cen}}^{\mathrm{H}} \mathbf{H}_{\text{cen}})^{-1} \mathbf{H}_{\text{cen}}^{\mathrm{H}} \mathbf{y}_{\text{cen}}. \tag{3}$$

Obviously, the information aggregation in the central node through the network requires routing protocols and communication effort, which is a big problem especially for a large-scale network. Hence, we prefer to solve (2) in a distributed way over the network and still achieve the central solution among all the nodes eventually.

To realize the distributed estimation, the centralized LS problem in (2) can be decomposed into several parallel local estimation problems by imposing a set of consensus constraints $\mathbf{x}_j = \mathbf{x}_i, \forall j \in \mathcal{J}, \forall i \in \mathcal{N}_j$ which enforce an agreement on the estimates. Thus, we reformulate (2) into an constrained optimization problem as

$$\min_{\{\mathbf{x}_j | j \in \mathcal{J}\}} \ \frac{1}{2} \sum_{j=1}^{J} \|\mathbf{y}_j - \mathbf{H}_j \mathbf{x}_j\|^2 \qquad \text{s.t.} \quad \mathbf{x}_j = \mathbf{x}_i, \ \forall j \in \mathcal{J}, \forall i \in \mathcal{N}_j, \tag{4}$$

where $\mathbf{x}_j \in \mathbb{C}^{N_{\mathrm{I}}}$ is the local estimate on source data $\boldsymbol{x}$ at node j. (4) can be solved in a distributed way using the distributed consensus-based estimation (DiCE) algorithm [11], where auxiliary variables $\mathbf{z}_j \in \mathbb{C}^{N_{\mathrm{I}}}$ per node j are introduced to decouple the consensus constraints, i.e., $\mathbf{x}_j = \mathbf{z}_j$ and $\mathbf{x}_j = \mathbf{z}_i, j \in \mathcal{J}, i \in \mathcal{N}_j$, and to facilitate parallel processing. When we define a set $\mathcal{N}_j' = \mathcal{N}_j \cup \{j\}$, all consensus constraints of DiCE can be written as $\mathbf{x}_j = \mathbf{z}_i, i \in \mathcal{N}_j'$. Utilizing the alternating direction method of multipliers (ADMM) [20], the DiCE algorithm provides the update equations of local estimate $\mathbf{x}_j^{k+1}$ and variable $\mathbf{z}_j^{k+1}$ by minimizing the Lagrange cost function of (4) w.r.t. $\mathbf{x}_j$ and $\mathbf{z}_j$ in iteration $k+1$:

$$\mathbf{x}_j^{k+1} = \min_{\mathbf{x}_j} \mathcal{L}_j(\mathbf{x}_j, \mathbf{z}_i^k, \boldsymbol{\alpha}_{ji}^k), \tag{5}$$

$$\mathbf{z}_j^{k+1} = \min_{\mathbf{z}_j} \mathcal{L}_j'(\mathbf{z}_j, \mathbf{x}_i^{k+1}, \boldsymbol{\alpha}_{ij}^k), \tag{6}$$

with the update of Lagrange multipliers $\boldsymbol{\alpha}_{ji} \in \mathbb{C}^{N_{\mathrm{I}}}$

$$\boldsymbol{\alpha}_{ij}^{k+1} = \boldsymbol{\alpha}_{ij}^k - \frac{1}{\mu}(\mathbf{x}_j^{k+1} - \mathbf{z}_i^{k+1}), \tag{7}$$

where μ is a positive step size parameter. Obviously, the update of $\mathbf{x}_j^{k+1}$ and $\mathbf{z}_j^{k+1}$ need the information on $\mathbf{z}_i^k$ and $\mathbf{x}_i^{k+1}$, respectively. Hence, after the update (5) each node $j \in \mathcal{J}$ transmits $\mathbf{x}_j^{k+1}$ to its neighboring nodes $i \in \mathcal{N}_j$ and then update $\mathbf{z}_j^{k+1}$ based on (6). After that, each node $j \in \mathcal{J}$ transmits $\mathbf{z}_j^{k+1}$ to its neighboring nodes $i \in \mathcal{N}_j$ for the calculation of estimate in the next iteration. In order to reduce the communication overhead, we will exploit the VC strategy in the following.

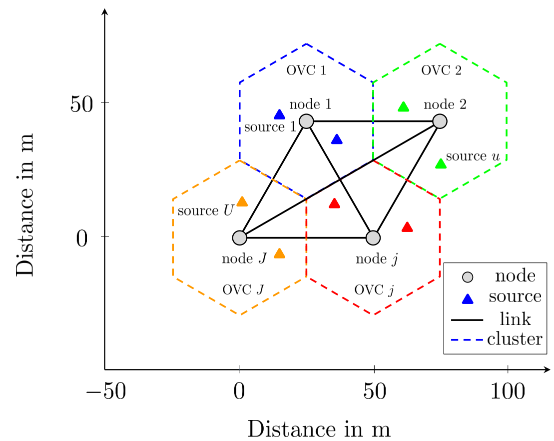

Fig. 1. A network of J nodes with J original virtual clusters. All nodes cooperate through inter-node links to detect the common messages from U sources located in different virtual clusters.

III. Virtual Clustering Strategy for Distributed Consensus-Based Estimation

The VC strategy is defined to classify the estimates into different sets. Thus, the consensus constraints in (4) can be split into several partial constraints with smaller dimension, which enables the reduction of unnecessary transmission.

A. Virtual clustering for partial consensus constraint

In the VC strategy, we assume that J nodes are uniformly distributed in an area where U sources exist. We define an original virtual cluster (OVC) around each node to cover one or more sources close to it. Fig. 1 illustrates an example of regular virtual clustering setting, where J hexagonal OVCs are defined for J nodes in a fully meshed topology. We further define that a source $u \in \{1, \cdots, U\}$ located in one specific OVC $\kappa \in \{1, .., J\}$ belongs to set $\mathcal{V}_\kappa$, where κ denotes cluster index. More specifically, if source u is close to node j, it is set to be located in the corresponding OVC j. Thus, based on the virtual clustering classification, the estimate $\mathbf{x}_j$ of node j can be classified into a set of sub-vectors $\{\mathbf{x}_{j,1}, \mathbf{x}_{j,2}, ..., \mathbf{x}_{j,J}\}$, where each $\mathbf{x}_{j,\kappa}$ denotes estimate on all sources $u \in \mathcal{V}_\kappa$. Note that, in this paper, the first index of vector $\mathbf{x}_{j,\kappa}$ denotes different nodes j and the second index indicates different clusters κ. Correspondingly, the consensus constraints $\mathbf{x}_j = \mathbf{x}_i, \forall j \in \mathcal{J}, \forall i \in \mathcal{N}_j$ in (4) can be rewritten with respect to the sub-vector $\mathbf{x}_{j,\kappa}$ leading to partial consensus constraints, i.e., $\mathbf{x}_{j,\kappa} = \mathbf{x}_{i,\kappa}, \kappa \in \{j, i\}$, between node j and its neighboring nodes $i \in \mathcal{N}_j$. For example, if we set $J = 4$ in Fig. 1, the estimate $\mathbf{x}_j$ of node j can be equivalently expressed by four stacked sub-vectors $[\mathbf{x}_{j,1}^{\mathrm{T}}, \mathbf{x}_{j,2}^{\mathrm{T}}, \mathbf{x}_{j,3}^{\mathrm{T}}, \mathbf{x}_{j,4}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{C}^{N_{\mathrm{I}}}$. Thus, the overall consensus constraint can be split into partial constraints as

$$\mathbf{x}_1 = \mathbf{x}_2 = \mathbf{x}_3 = \mathbf{x}_4 \Rightarrow \begin{cases} \mathbf{x}_{1,1} = \mathbf{x}_{2,1} = \mathbf{x}_{3,1} = \mathbf{x}_{4,1}, \\ \mathbf{x}_{1,2} = \mathbf{x}_{2,2} = \mathbf{x}_{3,2} = \mathbf{x}_{4,2}, \\ \mathbf{x}_{1,3} = \mathbf{x}_{2,3} = \mathbf{x}_{3,3} = \mathbf{x}_{4,3}, \\ \mathbf{x}_{1,4} = \mathbf{x}_{2,4} = \mathbf{x}_{3,4} = \mathbf{x}_{4,4}. \end{cases} \tag{8}$$

As mentioned in the DiCE algorithm, the exchanged vectors between nodes highly depend on the corresponding constraints. By constructing the partial consensus constraints, the exchanged local information in every iteration does not need to be the entire vector as before but sub-vector. In this way, it is possible and flexible to abandon some unnecessary information for transmission, i.e., transmit the reduced size of data, in order to achieve communication cost reduction.

B. Virtual data selection matrix

To describe each specific data sub-vector, a VDS matrix $\mathbf{C}_{j\kappa}$ is defined at each node $j \in \mathcal{J}$ to select the partial data from node j as well as neighboring nodes $i \in \mathcal{N}_j$ related to the source in cluster $\kappa \in \{j, i\}$ as $\mathbf{x}_{j,\kappa} = \mathbf{C}_{j\kappa}\mathbf{x}_j$ and $\mathbf{x}_{i,\kappa} = \mathbf{C}_{j\kappa}\mathbf{x}_i$. Referring to the example with full mesh topology in Fig. 1, the sub-vectors with respect to each cluster $\kappa \in \{1, \cdots, 4\}$ can be obtained by VDS matrices $\mathbf{C}_{j1} = [\mathbf{I}, \mathbf{0}, \mathbf{0}, \mathbf{0}], \mathbf{C}_{j2} = [\mathbf{0}, \mathbf{I}, \mathbf{0}, \mathbf{0}], \mathbf{C}_{j3} = [\mathbf{0}, \mathbf{0}, \mathbf{I}, \mathbf{0}], \mathbf{C}_{j4} = [\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{I}], j = 1, .., 4$. Note that the network topology will influence the size of the VDS matrices. How to determine the VDS matrices for a random topology will be discussed in Section IV.

C. Virtual clustering based DiCE algorithm

As mentioned in Section II, the DiCE algorithm is developed to solve the constrained optimization problem (4). We apply the VC strategy to the DiCE algorithm resulting in the VC-DiCE algorithm. First of all, we combine the VDS matrices with the constraints of DiCE and the corresponding optimization problem with consensus sub-constraints can be written as

$$\begin{aligned} \min_{\{\mathbf{x}_j | j \in \mathcal{J}\}} \; & \frac{1}{2}\sum_{j=1}^{J} \|\mathbf{y}_j - \mathbf{H}_j\mathbf{x}_j\|^2 \\ \text{s.t.} \quad & \mathbf{C}_{ji}\mathbf{x}_j = \mathbf{C}_{ji}\mathbf{z}_i, \\ & \mathbf{C}_{jj}\mathbf{x}_j = \mathbf{C}_{jj}\mathbf{z}_i, \; \forall j \in \mathcal{J}, \forall i \in \mathcal{N}_j'. \end{aligned} \tag{9}$$

By applying the ADMM approach for solving the above problem, the Lagrangian multipliers $\boldsymbol{\lambda}_{ji}$ and $\boldsymbol{\nu}_{ji}$ have to be introduced which are associated with the constraint in (9). Note that, the dimensions of $\boldsymbol{\lambda}_{ji}$ and $\boldsymbol{\nu}_{ji}$ are initialized according to the VDS matrices $\mathbf{C}_{ji}$ and $\mathbf{C}_{jj}$, respectively. The corresponding augmented Lagrangian cost function $\mathcal{L}(\,\cdot\,)$ over all nodes can be obtained and decomposed into J local sub-functions $\mathcal{L}_j(\mathbf{x}_j, \,\cdot\,)$ w.r.t. $\mathbf{x}_j$:

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}, \boldsymbol{\nu}) &= \frac{1}{2}\sum_{j=1}^{J} \|\mathbf{y}_j - \mathbf{H}_j\mathbf{x}_j\|^2 \\ & - \sum_{j=1}^{J}\sum_{i \in \mathcal{N}_j'} \left[\boldsymbol{\lambda}_{ji}^{\mathrm{T}}(\mathbf{C}_{ji}\mathbf{x}_j - \mathbf{C}_{ji}\mathbf{z}_i) + \boldsymbol{\nu}_{ji}^{\mathrm{T}}(\mathbf{C}_{jj}\mathbf{x}_j - \mathbf{C}_{jj}\mathbf{z}_i)\right] \\ & + \sum_{j=1}^{J}\sum_{i \in \mathcal{N}_j'} \frac{1}{2\mu}\left[\|\mathbf{C}_{ji}\mathbf{x}_j - \mathbf{C}_{ji}\mathbf{z}_i\|^2 + \|\mathbf{C}_{jj}\mathbf{x}_j - \mathbf{C}_{jj}\mathbf{z}_i\|^2\right] \\ & = \sum_{j=1}^{J} \mathcal{L}_j(\mathbf{x}_j, \mathbf{z}, \boldsymbol{\lambda}, \boldsymbol{\nu}), \end{aligned} \tag{10}$$

where μ is a positive step size parameter. The local cost function $\mathcal{L}_j(\mathbf{x}_j, \,\cdot\,)$ can be minimized w.r.t. $\mathbf{x}_j$ at each node $j \in \mathcal{J}$ individually. Given $\mathbf{z}^k$, $\boldsymbol{\lambda}^k$ and $\boldsymbol{\nu}^k$, we firstly calculate the update of $\mathbf{x}_j^{k+1}$ by minimizing $\mathcal{L}_j(\mathbf{x}_j, \mathbf{z}^k, \boldsymbol{\lambda}^k, \boldsymbol{\nu}^k)$, i.e., $\partial\mathcal{L}_j(\mathbf{x}_j, \mathbf{z}^k, \boldsymbol{\lambda}^k, \boldsymbol{\nu}^k)/\partial\mathbf{x}_j = 0$ resulting in

$$\begin{aligned} \mathbf{x}_j^{k+1} = & \left[\mathbf{H}_j^{\mathrm{H}}\mathbf{H}_j + \frac{1}{\mu}\sum_{i \in \mathcal{N}_j'}\left(\mathbf{C}_{jj}^{\mathrm{T}}\mathbf{C}_{jj} + \mathbf{C}_{ji}^{\mathrm{T}}\mathbf{C}_{ji}\right)\right]^{-1} \\ & \left[\mathbf{H}_j^{\mathrm{H}}\mathbf{y}_j + \sum_{i \in \mathcal{N}_j'}\left(\mathbf{C}_{ji}^{\mathrm{T}}\boldsymbol{\lambda}_{ji}^k + \mathbf{C}_{jj}^{\mathrm{T}}\boldsymbol{\nu}_{ji}^k\right)\right. \\ & \left. + \sum_{i \in \mathcal{N}_j'}\frac{1}{\mu}\left(\mathbf{C}_{ji}^{\mathrm{T}}\mathbf{C}_{ji}\mathbf{z}_i^k + \mathbf{C}_{jj}^{\mathrm{T}}\mathbf{C}_{jj}\mathbf{z}_i^k\right)\right]. \end{aligned} \tag{11}$$

After the estimate $\mathbf{x}_j^{k+1}$ is updated by each node $j \in \mathcal{J}$, partial vectors $\mathbf{C}_{ij}\mathbf{x}_j^{k+1}$ and $\mathbf{C}_{ii}\mathbf{x}_j^{k+1}$ will be transmitted from node j to its neighboring nodes $i \in \mathcal{N}_j$. Next, following the derivation in [11], the overall cost function in (10) can be rewritten into the sum of local cost function $\mathcal{L}_j'(\mathbf{z}_j, \,\cdot\,)$ w.r.t. $\mathbf{z}_j$. Then, the update equation of $\mathbf{z}_j^{k+1}$ is calculated by setting $\partial\mathcal{L}_j'(\mathbf{z}_j, \mathbf{x}^{k+1}, \boldsymbol{\lambda}^k, \boldsymbol{\nu}^k)/\partial\mathbf{z}_j = 0$, and we obtain

Algorithm 1 VC-DiCE Algorithm

1: Initialize $\boldsymbol{\lambda}_{ji}^0 = \mathbf{0}$, $\boldsymbol{\nu}_{ji}^0 = \mathbf{0}$, $\mathbf{x}_j^0 = \mathbf{z}_j^0 = \mathbf{0}$ for all $j \in \mathcal{J}, i \in \mathcal{N}_j'$
2: **for** $k = 0, \cdots, K$, each node j **do**
3: update the local estimate $\mathbf{x}_j^{k+1}$ according to (11), and transmit the partial estimates $\mathbf{C}_{ii}\mathbf{x}_j^{k+1}$, $\mathbf{C}_{ij}\mathbf{x}_j^{k+1}$ to its neighboring nodes $i \in \mathcal{N}_j$
4: update the auxiliary variable $\mathbf{z}_j^{k+1}$ according to (12), and transmit the partial variables $\mathbf{C}_{ii}\mathbf{z}_j^{k+1}, \mathbf{C}_{ij}\mathbf{z}_j^{k+1}$ to neighboring nodes $i \in \mathcal{N}_j$
5: update the multipliers $\boldsymbol{\lambda}_{ji}^{k+1}$ and $\boldsymbol{\nu}_{ji}^{k+1}$ according to (13) and (14), respectively, and transmit the multipliers $\boldsymbol{\lambda}_{ji}^{k+1}, \boldsymbol{\nu}_{ji}^{k+1}$ to its neighboring nodes $i \in \mathcal{N}_j$
6: **end for**

$$\mathbf{z}_j^{k+1} = \left[\frac{1}{\mu}\sum_{i\in\mathcal{N}_j'}\left(\mathbf{C}_{jj}^{\mathrm{T}}\mathbf{C}_{jj} + \mathbf{C}_{ji}^{\mathrm{T}}\mathbf{C}_{ji}\right)\right]^{-1} \left[-\sum_{i\in\mathcal{N}_j'}\left(\mathbf{C}_{jj}^{\mathrm{T}}\boldsymbol{\lambda}_{ij}^k + \mathbf{C}_{ji}^{\mathrm{T}}\boldsymbol{\nu}_{ij}^k\right) + \sum_{i\in\mathcal{N}_j'}\frac{1}{\mu}\left(\mathbf{C}_{jj}^{\mathrm{T}}\mathbf{C}_{jj}\mathbf{x}_i^{k+1} + \mathbf{C}_{ji}^{\mathrm{T}}\mathbf{C}_{ji}\mathbf{x}_i^{k+1}\right)\right]. \quad (12)$$

Once the new update of $\mathbf{z}_j^{k+1}$ is achieved, each node j shares partial vectors $\mathbf{C}_{ij}\mathbf{z}_j^{k+1}$ and $\mathbf{C}_{ii}\mathbf{z}_j^{k+1}$ with its neighboring nodes $i \in \mathcal{N}_j$. Finally, with the new update of $\mathbf{x}_j^{k+1}$ and $\mathbf{z}_j^{k+1}$, each node j can update the Lagrangian multipliers $\boldsymbol{\lambda}_{ji}$ and $\boldsymbol{\nu}_{ji}$ locally, following the update equations:

$$\boldsymbol{\lambda}_{ji}^{k+1} = \boldsymbol{\lambda}_{ji}^k - \frac{1}{\mu}\left(\mathbf{C}_{ji}\mathbf{x}_j^{k+1} - \mathbf{C}_{ji}\mathbf{z}_i^{k+1}\right), \quad (13)$$

$$\boldsymbol{\nu}_{ji}^{k+1} = \boldsymbol{\nu}_{ji}^k - \frac{1}{\mu}\left(\mathbf{C}_{jj}\mathbf{x}_j^{k+1} - \mathbf{C}_{jj}\mathbf{z}_i^{k+1}\right). \quad (14)$$

Observing the update equations (11)-(14) of the VC-DiCE algorithm, the update of $\mathbf{x}_j, \mathbf{z}_j, \boldsymbol{\lambda}_{ji}, \boldsymbol{\nu}_{ji}$ on each node j only need the sub-vectors $\mathbf{x}_{i,i} = \mathbf{C}_{ji}\mathbf{x}_i$, $\mathbf{x}_{i,j} = \mathbf{C}_{jj}\mathbf{x}_i$, $\mathbf{z}_{i,i} = \mathbf{C}_{ji}\mathbf{z}_i$, $\mathbf{z}_{i,j} = \mathbf{C}_{jj}\mathbf{z}_i$ selected by the VDS matrices $\mathbf{C}_{j\kappa}$, $\kappa \in \{j, i\}$ and multipliers $\boldsymbol{\lambda}_{ij}, \boldsymbol{\nu}_{ij}$ with reduced dimension received from the neighboring nodes i via inter-node links. This is quite different from the DiCE algorithm where the entire vectors are exchanged between neighboring nodes. Obviously, by partial transmission of vectors between neighbors, VC-DiCE can reduce the inter-node communication overhead compared to DiCE. The transmitted sub-vectors with reduced dimensions are determined by some VDS matrices which do not have full column rank, i.e., rank$(\mathbf{C}_{j\kappa}) < N_{\mathrm{I}}$, $j \in \mathcal{J}, \kappa \in \{j, i\}, i \in \mathcal{N}_j$. The whole procedure of the VC-DiCE algorithm is summarized in Algorithm 1.

IV. Methods to Determine the VDS Matrix

The key issue of VC strategy applied to distributed consensus-based estimation is the determination of the VDS matrices $\mathbf{C}_{j\kappa}$, $j \in \mathcal{J}, \kappa \in \{j, i\}, i \in \mathcal{N}_j$. For different network topologies, the VDS matrices are also different. In this section, two methods to determine the VDS matrices for arbitrary network topologies will be introduced. First of all, we define a matrix $\mathbf{A} \in \mathbb{R}^{J\times J}$ as the adjacency matrix of the cooperative network with all diagonal elements to be 0. Besides, $\mathbf{A}_{ij} = 1$ stands for node i and node j are connected directly and $\mathbf{A}_{ij} = 0$ means unconnected otherwise. Both methods are developed under the assumption that each node has knowledge of $\mathbf{A}$. In addition, we also assume that the link between two nodes is bidirectional, which implies that $\mathbf{x}_{j,i} = \mathbf{x}_{i,i}$ and $\mathbf{x}_{i,i} = \mathbf{x}_{j,i}$ are two different pair-wise constraints.

A. Method 1

To ensure that the algorithm using the VC strategy converges to the central solution, the design of the VDS matrices $\mathbf{C}_{j\kappa}$, $\kappa \in \{j, i\}$ should make the partial constraints in (9) finally reach the condition $\mathbf{x}_{j,i} = \mathbf{x}_{i,i}$, $\forall j, i \in \mathcal{J}$ w.r.t. each OVC. In the existing consensus-based algorithms such as DiCE [11] and ALCE [14], at each node j, the consensus constraints $\mathbf{x}_j = \mathbf{x}_i$, $\forall j \in \mathcal{J}, \forall i \in \mathcal{N}_j$ are on the entire vector of estimate variables. For the nodes which do not have direct connections, the consensus represented by constraints of each node will be realized by multi-hops. More specifically, the intermediate nodes, as a 'bridge' to connect two nodes in an indirect connected link, deliver the entire vectors to neighboring nodes in order to achieve the consensus. However, for algorithm using VC strategy, partial consensus constraints are used, which indicates that only partial estimate is transmitted from one node to another. Which part of estimate transmitted by the intermediate nodes in the indirect link of two node is of vital importance to achieve the overall consensus for arbitrary network topologies.

Method 1 is developed based on finding the shortest path from one node to another with the least number of intermediate nodes in between. Thus, once the information is transmitted through the shortest path, it is unnecessary to enforce the node to deliver this information again through the other paths with more intermediate nodes. Here, we define the set $\mathcal{S}_{jm}$ containing the intermediate nodes in the shortest path from node j to another indirect connected node m. There are several well known algorithms such as Dijkstra's or Bellman-Ford algorithm [21] can be applied to calculate the shortest path from one node to another. To obtain the VDS matrices $\mathbf{C}_{j\kappa}$ at node j, two ***Rules*** must be followed:

Rule 1: *each node $j \in \mathcal{J}$ must transmit the estimate $\mathbf{x}_{j,j}$ w.r.t. OVC j to its neighbors.*
Rule 2: *each intermediate node $n \in \mathcal{S}_{jm}$ must transmit the partial estimate $\mathbf{x}_{n,j}$ w.r.t. OVC j to neighbors.*

The reason to follow ***Rule 1*** is the path-loss effect in practice is taken into consideration in our work. According to the definition of OVC, the range of OVC j will cover the sources close to node j. On the other hand, each node j observes the messages from all sources located in all OVCs. Hence, each node j obtains a better observation w.r.t. the sources

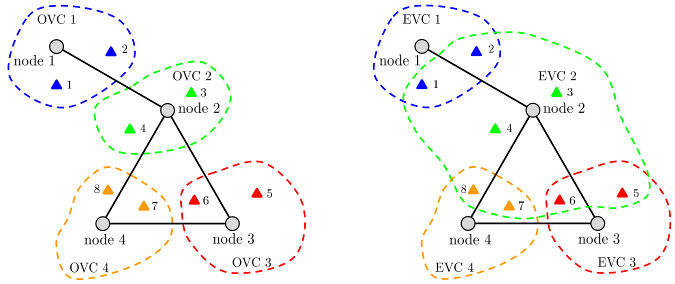

Fig. 2. An network example using Method 1: there are 4 nodes with 4 original virtual clusters (left) and 4 extended virtual clusters (right) after processing of Method 1

in OVC j due to lower influence of path-loss, compared to the observation on the sources in the other OVCs. The corresponding estimate on this part of source data at node j is $\mathbf{x}_{j,j}$. Intuitively, the spreading of the estimate $\mathbf{x}_{j,j}$ in the network will improve the performance of our VC-DiCE algorithm.

Following ***Rule 2***, the partial estimate $\mathbf{x}_{j,j}$ is spread from node j to all the other indirect connected nodes in the network. Correspondingly, we assume that the OVC area of each intermediate node $n \in \mathcal{S}_{jm}$ is extended resulting in an extended virtual cluster (EVC) which covers additional sources located in OVC j. If the OVC of one node does not change after processing, we directly enforce the EVC of this node to be identical to its OVC. In this way, for all nodes $j \in \mathcal{J}$ we can obtain the VDS matrices $\mathbf{C}_{j\kappa}$ which select partial transmitted data related to EVC $\kappa \in \{j, i\}$. As defined before, the set $\mathcal{V}_\kappa$ contains the sources in OVC. Now, we further define a set $\mathcal{V}_\kappa^{'}$ containing the sources in EVC.

Without of loss of generality, an example of a random network topology is shown in Fig. 2. Following Method 1, the extended sets regarding to sources in EVC $\kappa = 1, .., 4$ are $\mathcal{V}_1^{'} = \{1, 2\}, \mathcal{V}_2^{'} = \{1, 2, 3, 4, 5, 6, 7, 8\}, \mathcal{V}_3^{'} = \{5, 6\}, \mathcal{V}_4^{'} = \{7, 8\}$. Obviously, only the set of OVC 2 are extended, because in this example node 2 is an intermediate node in all potential shortest paths. Thus, the determined VDS matrices are $\mathbf{C}_{11} = \mathbf{C}_{21} = [\mathbf{I}, \mathbf{0}, \mathbf{0}, \mathbf{0}], \mathbf{C}_{12} = \mathbf{C}_{22} = \mathbf{C}_{32} = \mathbf{C}_{42} = \text{diag}(\mathbf{I}, \mathbf{I}, \mathbf{I}, \mathbf{I}), \mathbf{C}_{23} = \mathbf{C}_{33} = \mathbf{C}_{43} = [\mathbf{0}, \mathbf{0}, \mathbf{I}, \mathbf{0}], \mathbf{C}_{24} = \mathbf{C}_{34} = \mathbf{C}_{44} = [\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{I}]$, of which some matrices do not have full column rank. Hence, sub-vectors with reduced dimensions will be exchanged between neighbors resulting in the reduction of communication overhead using Method 1. Then we summarize the whole procedure explicitly in the processing steps of Method 1.

However, some redundancy still remains in the communication after processing of Method 1. To further reduce communication overhead, another method is exploited to calculate the VDS matrices.

B. Method 2

Recalling the general consensus constraints in (4), based on the overall knowledge of the adjacency, all constraints can also

Processing Steps of Method 1

Set $\mathcal{V}_\kappa^{'} = \mathcal{V}_\kappa$ for all OVC $\kappa \in \{1, .., J\}$
for $j = 1, \cdots, J$ **do**
  for $m \in \{1, \cdots, J | m \neq j\}$ **do**
    find the shortest path from node j to m based on adjacency matrix $\mathbf{A}$, save the intermediate nodes in set the $\mathcal{S}_{jm}$
    for all intermediate node $n \in \mathcal{S}_{jm}$ **do**
      extend the corresponding virtual cluster to EVC: $\mathcal{V}_n^{'} \leftarrow \mathcal{V}_n^{'} \cup \mathcal{V}_j$
    end for
  end for
end for
for all $j \in \{1, ..., J\}$ **do**
  determine the $\mathbf{C}_{j\kappa}$ which can select partial transmit data related to sources in $\mathcal{V}_\kappa^{'}$, $\kappa \in \{j, i\}$, $i \in \mathcal{N}_j$
end for

be rewritten into a centralized form as

$$\mathbf{B}\mathbf{x}_\text{s} = \mathbf{0} \tag{15}$$

where vector $\mathbf{x}_\text{s} = [\mathbf{x}_{1,1}^\text{T}, .., \mathbf{x}_{1,J}^\text{T}, ..., \mathbf{x}_{J,1}^\text{T}, .., \mathbf{x}_{J,J}^\text{T}]^\text{T} \in \mathbb{C}^{JN_\text{I}}$ is a stacked segmented estimate from all J nodes w.r.t. all OVC $\kappa \in \{1, .., J\}$ and $\mathbf{B} \in \mathbb{R}^{2|\mathcal{E}|N_\text{I} \times JN_\text{I}}$ is a block matrix containing block elements $\mathbf{I}$, $-\mathbf{I}$ and $\mathbf{0}$. Here, $|\mathcal{E}|$ is the number of edges of the network graph.

To fully remove the redundancy in constraint of (15), Method 2 is developed based on linear independence of constraint qualification (LICQ) [22], [23]. It holds at any feasible point of optimization variable if the set of gradients of the active constraints is linear independent [22]. LICQ can make sure that the set of constraints is well-defined without any redundancy. In our case, Method 2 should also follow the same ***Rule 1*** considering the influence of path-loss. Hence, the corresponding sub-constraint pairs always exist and should be reserved in (15). Then we reduce all the redundancy in the rest part of (15).

To implement Method 2, we introduce another matrix $\mathbf{P}$ with elements $1, -1$ and 0, which has the similar form to matrix $\mathbf{B}$[1]. Now, each row of $\mathbf{P}$ indicates the gradient of a specific sub-constraint, i.e., $\mathbf{x}_{j,\kappa} = \mathbf{x}_{i,\kappa}$, with specific j, i and OVC κ. Following ***Rule 1***, the rows related to the pair-wise sub-constraints $\mathbf{x}_{j,j} = \mathbf{x}_{i,j}, \mathbf{x}_{i,j} = \mathbf{x}_{j,j}, \forall j \in \mathcal{J}, i \in \mathcal{N}_j$ should always be retained in $\mathbf{P}$. In addition, for each specific j and i, the corresponding two rows in $\mathbf{P}$ which indicate pair-wise constraints, e.g., $\mathbf{x}_{1,1} = \mathbf{x}_{2,1}, \mathbf{x}_{2,1} = \mathbf{x}_{1,1}$, are regarded as linear independent in this work. Then the task is to remove the linear dependent rows in the rest part of matrix $\mathbf{P}$, which is equivalent to abandon all the redundant sub-constraints with respect to OVCs. In fact, the consensus on estimate expressed by the redundant sub-constraints can be achieved

[1] We assume that the dimension of the sub-vector $\mathbf{x}_{j,\kappa}^\text{T}$ of node j w.r.t. OVC κ is ignored, i.e., $\mathbf{x}_{j,\kappa}^\text{T}$ is considered as an element in $\mathbf{x}_\text{s}$. Thus, matrix $\mathbf{B}$ collapses into matrix $\mathbf{P}$ with elements $1, -1$ and 0.

Processing Steps of Method 2

1: Based on $\mathbf{A}$, rewrite a centralized form consensus constraint as (15)
2: Generate matrix $\mathbf{P}$ from $\mathbf{B}$
3: Retain the corresponding rows in $\mathbf{P}$ related to pair-wise constraints $\mathbf{x}_{j,j} = \mathbf{x}_{i,j}, \mathbf{x}_{i,j} = \mathbf{x}_{j,j}, \forall j \in \mathcal{J}, i \in \mathcal{N}_j$, w.r.t. OVC
4: Remove linear dependent rows in the rest of $\mathbf{P}$ and get a reduced form matrix $\mathbf{P}_{\text{re}}$
5: From $\mathbf{P}_{\text{re}}$, determine the corresponding $\mathbf{C}_{j\kappa}, \forall j \in \mathcal{J}, i \in \mathcal{N}_j, \kappa \in \{j, i\}$ by fixing $\mathbf{C}_{jj}$

via multiple remaining sub-constraints. We name the reduced matrix as $\mathbf{P}_{\text{re}}$ after removing redundancy in $\mathbf{P}$. From $\mathbf{P}_{\text{re}}$, we can get all reserved and non-redundant sub-constraints and further determine the corresponding VDS matrices $\mathbf{C}_{j\kappa}$, $\forall j \in \mathcal{J}, i \in \mathcal{N}_j, \kappa \in \{j, i\}$. Here, we fix $\mathbf{C}_{jj}$ as the j-th row of matrix $\text{diag}(\mathbf{I}, \mathbf{I}, \cdots, \mathbf{I})$ in order to select the partial estimate related to sources in OVC j following ***Rule 1***. Note that, if we fix $\mathbf{C}_{jj}$ in another form according to $\mathbf{P}_{\text{re}}$, the determined VDS matrices can be different, but the total number of surviving sub-constraints are the same and the number is related to the rank of matrix $\mathbf{P}$. Accordingly, we summarize the general process described above in the processing steps of Method 2.

Compared to Method 1, all unnecessary sub-constraints are abandoned, which leads to different VDS matrices $\mathbf{C}_{j\kappa}$ in Method 2. Considering the same topology in Fig. 2, we take $\mathbf{C}_{j\kappa}$ w.r.t. cluster 2 as an example and they are determined by Method 2 as $\mathbf{C}_{12} = [\mathbf{0}, \mathbf{I}, \mathbf{0}, \mathbf{0};\ \mathbf{0}, \mathbf{0}, \mathbf{I}, \mathbf{0};\ \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{I}]$, $\mathbf{C}_{22} = [\mathbf{0}, \mathbf{I}, \mathbf{0}, \mathbf{0}]$ and $\mathbf{C}_{32} = \mathbf{C}_{42} = [\mathbf{I}, \mathbf{0}, \mathbf{0}, \mathbf{0};\ \mathbf{0}, \mathbf{I}, \mathbf{0}, \mathbf{0}]$, which have smaller sizes compared to the full column rank matrices $\mathbf{C}_{j2}$ using Method 1. Thus, when all the redundancy in the consensus constraints are removed by Method 2, the sizes of VDS matrices and the corresponding transmitted sub-vectors can be further reduced leading to the least communication cost. For an time-invariant network topology, both processing of Method 1 and 2 can be performed off-line to determine the VDS matrices for VC-based algorithms.

V. Simulation Results

In the following, to show the performance of proposed virtual clustering strategy based algorithm, numerical simulation results are presented in this section. We assume that the channel between sources to nodes is i.i.d complex Gaussian and influenced by path-loss which is inversely proportional to the distance with loss exponent of 1. J nodes forms a network with connectivity ratio r defined as

$$r = \frac{|\mathcal{E}|}{0.5 \times J(J-1)}. \tag{16}$$

Different algorithms are evaluated by using the metric of the averaged mean square error between the local estimate and the true source message $\boldsymbol{x}$ defined by

$$\overline{\text{MSE}} = \frac{1}{J} \sum_{j=1}^{J} \text{E}\{\left\|\mathbf{x}_j^k - \boldsymbol{x}\right\|^2\}, \tag{17}$$

where the expectation is approximated by averaging over 1000 Monte Carlo experiments with random realizations of source message vectors $\boldsymbol{x}$, measurement noise vectors $\mathbf{n}_j$, channel coefficient matrices $\mathbf{H}_j$ and network topologies.

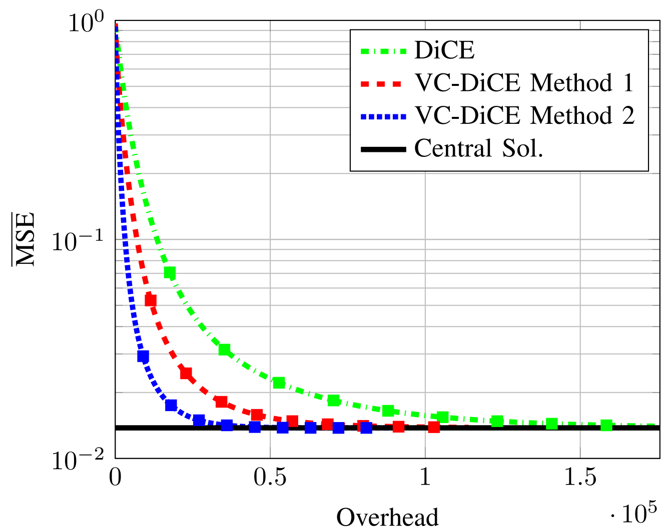

Fig. 3. $\overline{\text{MSE}}$ vs. number of total communication overhead, $J = U = 6$, $N_{\text{T}} = 1$, $N_{\text{R}} = 2$, the network is randomly connected, each marker represents an increment of 50 iterations

Fig. 3 depicts the $\overline{\text{MSE}}$ w.r.t. communication overhead for DiCE, VC-DiCE and the central solution (3). The communication overhead, defined as transmitted vector elements over node to node (N2N) links, increases along with the increasing number of iterations. The result shows that all algorithms asymptotically converge to the central solution after a sufficient number of iterations. For the same $\overline{\text{MSE}}$ performance level, VC-DiCE outperforms DiCE with a faster convergence rate and a considerable overhead reduction, which is in accord with our analysis in Section IV. Further, Method 2 can reduce more overhead than Method 1, because Method 2 enables the reduction of all unnecessary redundancy in consensus constraints providing the optimal communication cost for distributed consensus-based estimation.

Next, a systematic description of N2N communication overhead per iteration under different connectivity ratios r generated by DiCE and VC-DiCE is summarized in Fig. 4. We show 3 cases where $J = U$ varies, i.e., $J = U = 5, 10, 15$. For every specific r in each case, the overhead per iteration is averaged under randomly generated network topology by Monte Carlo simulation. The results show that VC-DiCE definitely outperforms DiCE. When r increases, the overhead produced by DiCE will also increase. However, in contrast, still comparatively small overhead will be produced by VC-DiCE with a large connectivity ratio. The reason is that in highly connected networks with large r, many possible paths

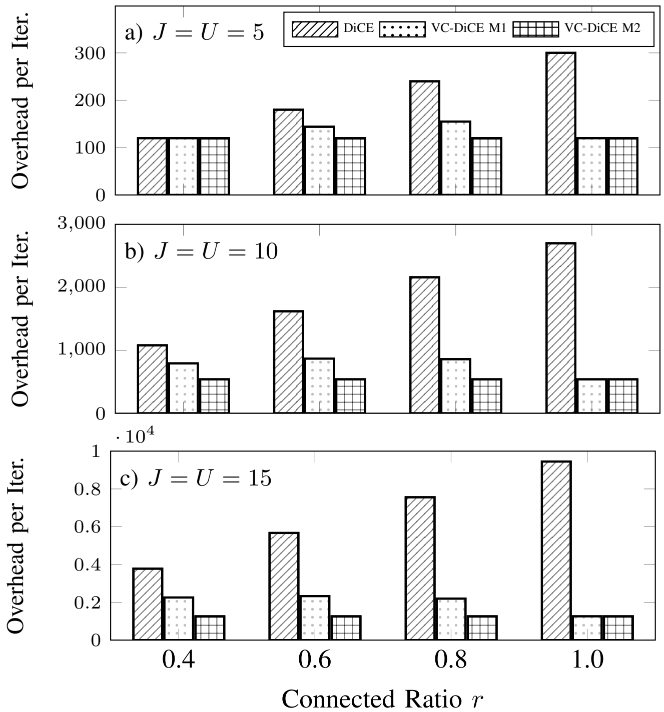

Fig. 4. Number of N2N communication overhead per iteration vs. varying connectivity ratio $r = 0.4, 0.6, 0.8, 1.0$, under 3 cases a) $J = U = 5$, b) $J = U = 10$, c) $J = U = 15$ generated by the DiCE algorithm (lines), the VC-DiCE algorithm Method 1 (dots) and the VC-DiCE algorithm Method 2 (grid), $N_{\mathrm{T}} = 1, N_{\mathrm{R}} = 2$.

exist from one node to another. This leads to huge unnecessary transmission (redundancy). By applying VC-DiCE, this redundant transmission can be abandoned and only small parts of estimate are transmitted by each node to its neighbors. Along with the increasing of r and network scale, VC-DiCE can save more overhead compared with DiCE. Especially for the case when $r = 1.0$ and $J = U = 15$, VC-DiCE has almost 87% reduction on overhead compared with DiCE. Moreover, independent from r , Method 2 can always offer an optimal number of communication cost per iteration and the number related to the rank of matrix $\mathbf{P}$, i.e., rank($\mathbf{P}$) does not change with r under an specific number of J. In general, Method 2 outperforms Method 1.

VI. Conclusion

In this paper, we develop the virtual clustering strategy to reduce inter-node communication overhead for the distributed consensus-based estimation. Two methods are proposed to determine the VDS matrices. We show that the VC-DiCE algorithm using either Method 1 or Method 2 achieves a significant reduction on communication overhead compared to the DiCE algorithm without sacrificing the performance. Moreover, Method 2 can further reduce the communication overhead compared to Method 1. It should be noticed that the virtual clustering strategy developed here is not limited to one specific algorithm but is also applicable to other distributed algorithms for consensus-based estimation.

References

[1] A. Swami, Q. Zhao, Y. Hong, and L. Tong, *Wireless sensor networks: signal processing and communications*. Wiley, Nov. 2007.

[2] C. Chen, S. Zhu, X. Guan, and X. Shen, *Wireless sensor networks: distributed consensus estimation*, 1st ed. Springer International Publishing, Dec. 2014.

[3] B.-S. Shin, H. Paul, and A. Dekorsy, "Distributed kernel least squares for nonlinear regression applied to sensor networks," in *24th European Signal Processing Conference (EUSIPCO 2016)*, Aug. 2016.

[4] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. of the 46th IEEE Conf. on Decision and Control*, 2007, pp. 5492–5498.

[5] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[6] A. Viseras, T. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino, "Decentralized multi-agent exploration with online-learning of gaussian processes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4222–4229.

[7] D. Wübben, H. Paul, B.-S. Shin, G. Xu, and A. Dekorsy, "Distributed consensus-based estimation for small cell cooperative networks," in *10th International Workshop on Broadband Wireless Access (BWA 2014), co-located with IEEE Globecom 2014*, Austin, TX, USA, 2014.

[8] Z. Chair and P. K. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-22, no. 1, pp. 98–101, Jan. 1986.

[9] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[10] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*, Apr. 2005, pp. 63–70.

[11] H. Paul, J. Fliege, and A. Dekorsy, "In-network-processing: distributed consensus-based linear estimation," *IEEE Communications Letters*, vol. 17, no. 1, pp. 59–62, 2013.

[12] H. Zhu, A. Cano, and G. Giannakis, "Distributed consensus-based demodulation: algorithms and error analysis," *IEEE Transactions on Wireless Communications*, vol. 9, no. 6, pp. 2044–2054, 2010.

[13] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[14] G. Xu, H. Paul, D. Wübben, and A. Dekorsy, "Distributed augmented lagrangian method for cooperative estimation in small cell networks," in *10th International ITG Conference on Systems, Communications and Coding (SCC 2015)*, Hamburg, Germany, Feb. 2015.

[15] Q. Ling, Y. Liu, W. Shi, and Z. Tian, "Weighted ADMM for fast decentralized network optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 22, pp. 5930–5942, 2016.

[16] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, "Distributed optimization with local domains: applications in MPC and network flows," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 2004–2009, 2015.

[17] R. Arablouei, S. Werner, Y.-F. Huang, and K. Doğançay, "Distributed least mean-square estimation with partial diffusion," *IEEE Transactions on Signal Processing*, vol. 62, no. 2, pp. 472–484, 2014.

[18] G. Xu, S. Wang, H. Paul, and A. Dekorsy, "Virtual clustering for distributed consensus-based estimation in cooperative networks," in *2017 IEEE 85th Vehicular Technology Conference (VTC17 Spring)*, Sydney, Australia, Jun 2017.

[19] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear estimation*. Prentice Hall Upper Saddle River, NJ, 2000, vol. 1.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[21] W. Stallings, *Data and computer communications*, 8th ed., 2006.

[22] M. V. Solodov, "Constraint qualifications," *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[23] G. Wachsmuth, "On LICQ and the uniqueness of Lagrange multipliers," *Operations Research Letters*, vol. 41, no. 1, pp. 78–80, 2013.