Towards Distributed Ensemble Clustering for Networked Sensing Systems: A Novel Geometric Approach

Hu Ding
Computer Science and Engineering
Michigan State University
huding@msu.edu

Lu Su
Computer Science and Engineering
SUNY at Buffalo
lusu@buffalo.edu

Jinhui Xu
Computer Science and Engineering
SUNY at Buffalo
jinhui@buffalo.edu

ABSTRACT

Given a set of different clustering solutions to a unified dataset, ensemble clustering is to aggregate them to yield a more accurate and robust solution. In recent years, ensemble clustering has been extensively studied and successfully applied to many areas. In this paper, we study a new variant of ensemble clustering, *distributed ensemble clustering*, motivated by the proliferation of networked sensing systems where communication is enabled between only connected nodes. Our goal is to aggregate the clustering solutions produced by the sensor nodes that observe the same set of objects. Different from traditional ensemble clustering problems, distributed ensemble clustering aims to achieve not only accurate clustering results, but also low communication cost among the nodes. To this end, we build a novel geometric optimization model that can be efficiently solved with theoretical quality guarantee. The proposed approach, bearing nice geometric properties, can be easily adapted to distributed settings without any sacrifice of clustering quality, and facilitates a dimension reduction procedure which can significantly reduce the communication complexity. We validate our approach on two benchmark datasets. Experimental results suggest that our approach can efficiently solve the distributed ensemble clustering problem, and outperform the baselines on both clustering accuracy and communication cost.

CCS Concepts

•Computer systems organization → Embedded and cyber-physical systems; •Networks → Network algorithms;

Keywords

Ensemble Clustering; Sensor Networks; Communication Efficient; Machine Learning; Computational Geometry

1. INTRODUCTION

MobiHoc'16, July 04-08, 2016, Paderborn, Germany
© 2016 ACM. ISBN 978-1-4503-4184-4/16/07…$15.00
DOI: http://dx.doi.org/10.1145/2942358.2942391

Clustering is a classical and fundamental problem [25, 14] in computer science with numerous applications in many different areas, such as machine learning, data mining, and networking. Its basic task is to group a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups.

The clustering problem has been extensively studied in the past and many elegant solutions have been obtained, with each of them adopting a different strategy. Due to its unavoidable bias, each individual clustering method often works well only for a certain type of datasets. Thus, a more accurate strategy, called *ensemble clustering* [18], is to aggregate a set of clustering solutions (of the same set of objects) generated by different clustering algorithms (or the same kind of clustering algorithms under different conditions as in sensor networks; see details later) into a unified solution. In this way, errors of each individual solution could cancel out each other and a more reliable and accurate solution can be achieved.

In this paper, we consider a new variant of ensemble clustering, called *distributed ensemble clustering*. Roughly speaking, we are given multiple clustering solutions of a set of objects, with each solution stored in a different node of a communication network with certain topology (*e.g.,* a tree), and the goal is to aggregate these solutions to form the "best" solution. The challenge of this problem stems from the fact that communication is allowed only between connected nodes. In other words, each node can talk only to its neighbors, and no global view of these clustering solutions is available.

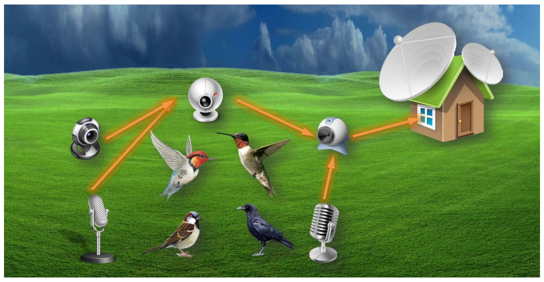


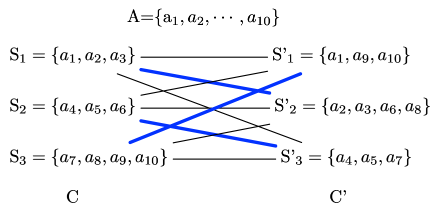
Figure 1: An example of networked sensing system

The problem is motivated by the proliferation of networked sensing systems where an object or event is usually observed by multiple sensors, as shown in Fig. 1. In many applications of sensor networks, such as target tracking [8], wildlife monitoring [15, 20, 21], assisted living [36, 40] and environment monitoring [10], one important task is to assign the observed object or event to one of several predefined classes. For example, consider the task of bird species classification as shown in Fig. 1, we can identify the species of the observed birds based on the video and audio information collected by the sensor nodes. To achieve this, a set of labelled training data are needed to derive the classification model. However, as pointed out in [34, 35], sensor networks are usually deployed in the remote, harsh, and sometimes even hostile locales, and thus it is extremely difficult to manually label a training set. Therefore, a more practical solution is to partition the observed objects into different groups using clustering algorithms without the use of any training data, and ask domain experts to manually label each group after the clustering results are delivered to the remote server operated by the end users [1]. To achieve this, a challenge has to be addressed. That is, due to various reasons such as poor sensor quality, lack of sensor calibration, background noise, and incomplete views of observations, each individual sensor node may not be very reliable and its clustering could be inaccurate. To improve sensing quality, it is desirable to use ensemble method to combine the clustering solutions produced locally by each individual sensor, as observations from different sensor nodes are often complementary to each other and aggregating them could lead to a better global view of the observed objects.

A straightforward approach is to gather the clustering solutions of all the sensor nodes to a sink node, and apply some traditional ensemble clustering algorithm on them. Although this can achieve optimized clustering accuracy, it would incur significant communication overhead, since communication is much more energy-expensive than computation, (*e.g.,* wireless transmission of a bit can require over 1000 times more energy than a single 32-bit computation [7]). Thus, a more reasonable strategy is to let intermediate nodes, instead of the sink, conduct clustering aggregations, since this could save substantial amount of communication energy that would otherwise be wasted forwarding all the clustering solutions to the sink. The required clustering and aggregation operations for normal sensor nodes could impose great challenges on traditional low-end sensing platforms such as Mica2 mote [24]. However, as many powerful sensing systems [38, 31, 19, 11] are rapidly emerging, we expect that such challenges could be eventually overcome, and thus accurate clustering solution and low communication cost can be simultaneously achieved.

Related work. Ensemble clustering was introduced by Strehl and Ghosh [32]. Following their seminal work, quite a few developments on ensemble clustering were proposed from both theory and application perspectives [16, 22, 2, 12, 33, 17]. For more detailed discussion about ensemble clustering, the reader is referred to a recent survey [18] . In the context of wireless sensor networks, Su *et al.* [34, 35] proposed a semi-supervised approach to combine the decisions of individual sensors. Their method assumes the availability of correctly labeled data and thus is quite different from the settings of our problem. The ensemble clustering problem studied in this paper falls under the umbrella of distributed machine learning. Driven by numerous big data applications, there are considerable growing interests in this

[1]Please note that due to the constraints of energy and bandwidth, there is usually no way to deliver all the raw data such as video and audio files to the remote server.



A={a$_1, a_2, \cdots, a_{10}$}
S$_1 = \{a_1, a_2, a_3\}$
S$_2 = \{a_4, a_5, a_6\}$
S$_3 = \{a_7, a_8, a_9, a_{10}\}$
S'$_1 = \{a_1, a_9, a_{10}\}$
S'$_2 = \{a_2, a_3, a_6, a_8\}$
S'$_3 = \{a_4, a_5, a_7\}$
C
C'

Figure 2: $\{S_1, S_2, S_3\}$ and $\{S'_1, S'_2, S'_3\}$ are two different clusterings of A; a bipartite graph is built with each edge connecting S_l and $S'_{l'}$ having the weight equal to their symmetric difference. The blue edges are the minimum weight bipartite matching.

area (such as distributed PCA) in recent years [3, 5, 13, 28]. Our investigation on distributed ensemble clustering could potentially become a new step along this line.

Our contributions. To our best knowledge, this is the first paper identifying and solving the distributed ensemble clustering problem. In this paper, we first transform ensemble clustering to a geometric optimization problem in high dimensional space based on some new insights to the problem (Section 3). We then develop a novel algorithm to solve the basic ensemble clustering problem with theoretical quality guarantee (Section 4.1), and further show that our algorithm can naturally be adapted to distributed settings (Section 4.2). In addition, since our model is formulated in Euclidean space, dimension reduction technique can be easily applied. This allows us to not only preserve the quality guarantee, but more importantly significantly reduce the communication cost (Section 4.3). Comparing to existing approaches for ensemble clustering, our proposed method has the following advantages. **(1).** Our algorithm can be efficiently realized in distributed settings without any sacrifice of quality, while existing methods are usually based on complicated graph models or optimization techniques (*e.g.,* semi-definite programming [33]) which are often challenging to be implemented in distributed settings. **(2).** To the best of our knowledge, this is the first combinatorial geometric approach for ensemble clustering. This novel design leads to significant reduction on communication cost and thus could potentially benefit a wide spectrum of applications.

2. PRELIMINARIES

In this section, we will introduce several definitions which will be used throughout this paper and the system overview for distributed ensemble clustering.

DEFINITION 1 (ENSEMBLE CLUSTERING). *Given a collection of objects $\mathcal{A} = \{a_1, a_2, \cdots, a_N\}$ and a set of clustering solutions $\{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_m\}$, where each $\mathcal{C}_j$ partitions $\mathcal{A}$ into $k \in \mathbb{Z}^+$ clusters[2], ensemble clustering is to find a new single clustering $\tilde{\mathcal{C}}$ for $\mathcal{A}$ so as to minimize its total differences to the m solutions*

$$\sum_{j=1}^{m} \Delta(\tilde{\mathcal{C}}, \mathcal{C}_j), \tag{1}$$

where the function Δ is defined in the following Definition 2.

[2]Note that if $\mathcal{C}_j$ has less than k clusters, we can always make up to k by adding dummy empty clusters.

We follow the idea in [4] to define the difference between two clusterings.

Definition 2 (Clustering Difference). *Let $\mathcal{C} = \{S_1, \cdots, S_k\}$ and $\mathcal{C}' = \{S'_1, \cdots, S'_k\}$ be two clusterings of $\mathcal{A}$, and $\mathcal{G}$ be a bipartite graph built between $\mathcal{C}$ and $\mathcal{C}'$ as follows: each of the two columns of $\mathcal{G}$ contains k vertices corresponding to the k clusters in $\mathcal{C}$ and $\mathcal{C}'$ respectively; for any pair of clusters $(S_l, S'_{l'})$ with $S_l \in \mathcal{C}$ and $S'_{l'} \in \mathcal{C}'$, there is an edge connecting their corresponding vertices in $\mathcal{G}$ with a weight equal to the size of their symmetric difference,* i.e., $|S_l \setminus S'_{l'}| + |S'_{l'} \setminus S_l|$. *Then, the difference of $\mathcal{C}$ and $\mathcal{C}'$, $\Delta(\mathcal{C}, \mathcal{C}')$, is the cost of the* ***minimum weight bipartite matching*** *of $\mathcal{G}$.*

In the above definition, the function Δ (*i.e.,* minimum weight bipartite matching) can be computed by using *Hungarian algorithm* in $O(k^3)$ time. Fig. 2 gives an example of clustering difference. The bipartite graph is built for $\mathcal{C}$ and $\mathcal{C}'$ (*e.g.,* the edge connecting S_1 and S'_2 has weight 3 which is equal to their symmetric difference), and the minimum weight bipartite matching is shown in blue, where $\Delta(\mathcal{C}, \mathcal{C}') = 3 + 2 + 3 = 8$.

Remark 1. *Note that Δ is a natural way of measuring clustering difference. If $\mathcal{C}$ is the ground truth, with simple calculation we know that $\Delta(\mathcal{C}, \mathcal{C}')$ always equals to two times of the number of wrongly clustered objects by $\mathcal{C}'$. See Appendix for details.*

2.1 Distributed Ensemble Clustering and System Overview

We consider the ensemble clustering problem in a distributed setting, where each clustering solution $\mathcal{C}_j$ is stored in a node (with a slight abuse of notation, for $1 \le j \le m$, we denote the corresponding node as $\mathcal{C}_j$ as well). These m nodes form a connected graph, and communication is restricted to the pairs of nodes which are connected by edges. For any arbitrary connected graph, we can always compute a *spanning tree*, denoted as $\mathbb{H}$, using breath first search method (see Fig. 3(a)). Throughout this paper, we will execute and analyze our algorithm only on $\mathbb{H}$, and show that the performance of our algorithm is independent of the topology of $\mathbb{H}$ (see detailed proof in Section 4.2). Note that the simplest communication model is that all nodes are connected to a central node, which actually form a tree with height one (Fig. 3(c)). However, in many scenarios, this model does not work well. For example, in wireless sensor networks [34, 35], one sensor can only transmit data to its neighborhoods according to the *unit disk graph* [27]. Thus, as a generalization, we assume that $\mathbb{H}$ is an arbitrary tree in this paper.

Definition 3 (Communication Complexity). *Given a tree $\mathbb{H}$ formed by a set of nodes with each of them storing a set of data, the communication complexity of an algorithm is the total amount of bits transmitted along all the edges of $\mathbb{H}$.*

For instance, in Fig. 3 (b), if the node a transmits 1 bit to the root, the total communication complexity is 2 since the data travel through two edges.

Furthermore, to avoid packet collision and overhearing during the process of clustering aggregation, we can employ some existing distributed aggregation scheduling algorithms such as [39]. Under such scheduling strategy, at any time slot only a subset of the sensor nodes are allowed to send packets and their transmissions do not interfere with each other. The wireless interfaces of the rest nodes are shut down so as to save the energy of idle listening.

Main challenges. There are two difficulties in solving distributed ensemble clustering. Firstly, ensemble clustering itself is a challenging combinatorial optimization problem. Its challenge mainly comes from that fact that we do not know which cluster should be matched to which cluster in the function Δ according to Definition 2, since $\tilde{C}$ is unknown in advance. In other words, we have to find the desired $\tilde{C}$ and minimize the total cost of bipartite matchings simultaneously.

Secondly, the distributed setting makes the problem even more challenging. To overcome this difficulty, one simple solution is to let all nodes send their data to the root and thus convert the problem to a centralized ensemble clustering problem. However, this could result in rather high communication cost (see the example in Fig. 3(d), where the nodes form a chain; if each node sends its data containing L bits to the root, the total communication cost is $\sum_{j=1}^{m-1} j * L = O(m^2 L)$). Note that some existing compressive sensing techniques for data gathering [30, 37, 23] are not appropriate to handle this issue, since they require that the data is sparse. However, the data in our ensemble clustering problem may not necessarily be sparse (see the detailed explanation at the end of Section 4.3). Another possible solution is to let each node compute a local solution for the data stored in its children and itself, and return the result to its parent; finally an aggregated solution can be obtained in the root. However, as illustrated in [34], this strategy cannot always yield a global optimal solution and could sacrifice quite a bit in accuracy. Thus, for distributed ensemble clustering, we need to minimize the communication cost while preserving the accuracy of the solution. In following sections, we will show that our approach yields a quality guaranteed solution with communication complexity linear in m and independent of the topology of $\mathbb{H}$.

3. OUR NEW FORMULATION FOR ENSEMBLE CLUSTERING

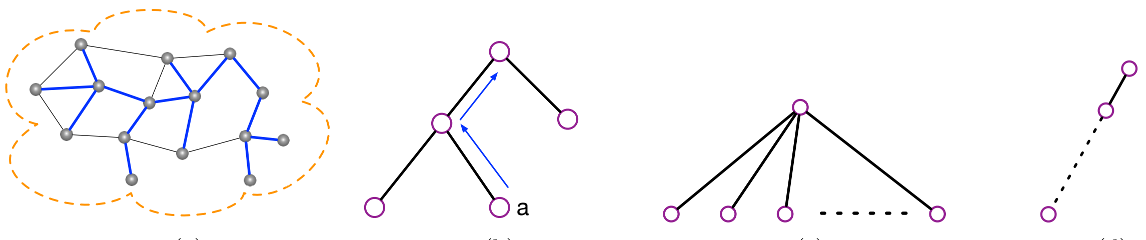
In this section, we focus on developing a new formulation for ensemble clustering, which transforms it to a geometric optimization problem in high dimensional Euclidean space. In Section 4, we will present an algorithm solving the geometric problem in both centralized and distributed settings.

Mapping. Following the notations in Definition 1, we map each clustering $\mathcal{C}_j$ to k-tuple points in $\mathbb{R}^N$, where N is the number of objects and each cluster corresponds to one point as follows. Each coordinate of $\mathbb{R}^N$ indicates the membership of one individual object in $\mathcal{A}$, that is, each cluster $S \in \mathcal{C}_j$ is mapped to a point

$$\begin{aligned} & p(S) = (x_1, x_2, \cdots, x_N), \quad \forall 1 \le i \le N, \\ & \textit{where} \quad x_i = 1 \ \textit{if} \ a_i \in S, \ \textit{else} \ x_i = 0. \end{aligned} \tag{2}$$

The following lemma shows that the difference of two clusters, *i.e.,* their symmetric difference according to Definition 2, can be naturally embedded into Euclidean space by the above mapping.

Lemma 1. *Given any two clusters S, $S' \subset \mathcal{A}$, the size of their symmetric difference $|S \setminus S'| + |S' \setminus S| = ||p(S) - p(S')||^2$.*



(a) (b) (c) (d)

Figure 3: (a) Wireless sensor network with each node representing a sensor; each edge indicates a link and the blue edges form a spanning tree. (b) Illustration of sending data from node a to the root of $\mathbb{H}$. (c) and (d) show two extreme cases for $\mathbb{H}$, all the nodes connecting to the root and the nodes forming a chain.

Proof. From formula (2), it is easy to know that

$$\begin{aligned} ||p(S) - p(S')||^2 &= |S \setminus S'| \times 1^2 + |S' \setminus S| \times (-1)^2 \\ &= |S \setminus S'| + |S' \setminus S|, \end{aligned} \quad (3)$$

which completes the proof. □

For convenience, we use $p(\mathcal{C}_j)$ to denote the mapped k-tuple of points in $\mathbb{R}^N$, and similar to Definition 2, we still use Δ to denote the cost of minimum weight bipartite matching between two k-tuples in $\mathbb{R}^N$, where each edge has the weight equal to their squared Euclidean distance (according to Lemma 1). We have the following theorem revealing the correctness of our new formulation.

Theorem 1. *Given an instance of ensemble clustering as in Definition 1, finding the optimal clustering solution $\tilde{\mathcal{C}}$ minimizing the objective function (1) is equivalent to finding a k-tuple of points $\mathcal{T} \subset \mathbb{R}^N$ such that $\sum_{j=1}^m \Delta(\mathcal{T}, p(\mathcal{C}_j))$ is minimized, where each point $t \in \mathcal{T}$ is a boolean vector,* i.e., *each coordinate has the value equal to either* 0 *or* 1, *and* $\sum_{t\in\mathcal{T}} t = \underbrace{(1, 1, \cdots, 1)}_{N}$.

Remark 2. *t is required to be a boolean vector since it should be able to be mapped back to a cluster (i.e., a subset) of $\mathcal{A}$, and $\sum_{t\in\mathcal{T}} t = (1, 1, \cdots, 1)$ guarantees that each object of $\mathcal{A}$ belongs to one and only one cluster.*

The combinatorial nature of requiring each point in $\mathcal{T}$ to be a boolean vector makes the problem quite challenging[3]. A natural idea is to relax this requirement, and see whether the result still makes sense. For ease of discussion, in the remaining parts of this paper we denote that the optimal k-tuple $\tilde{\mathcal{T}} = \{\tilde{t}_1, \cdots, \tilde{t}_k\}$ and $p(\mathcal{C}_j) = \{c^j_1, \cdots, c^j_k\}$ for each $1 \le j \le m$. Furthermore, the minimum weight bipartite matching between $\tilde{\mathcal{T}}$ and $p(\mathcal{C}_j)$ is denoted as δ_j, *i.e.*, $\Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j)) = \sum_{l=1}^k ||\tilde{t}_l - c^j_{\delta_j(l)}||^2$.

Lemma 2. *If each point of $\tilde{\mathcal{T}}$ is not restricted to be a boolean vector, $\sum_{l=1}^k \tilde{t}_l$ will be automatically guaranteed to be $\underbrace{(1, 1, \cdots, 1)}_{N}$, and each $\tilde{t}_l$ has non-negative value in each coordinate.*

[3]To the best of our knowledge, there is no prior work showing the hardness of ensemble clustering. A closely related problem, consensus clustering, is proved to be even APX-hard [2, 12] when the number of clusters k is not fixed.

Prior proving Lemma 2, we need the following lemma first.

Lemma 3. *If each point of $\tilde{\mathcal{T}}$ is not restricted to be a boolean vector, $\tilde{t}_l$ is the mean point of $\{c^j_{\delta_j(l)} \mid 1 \le j \le m\}$ for each $1 \le l \le k$.*

Proof. Note that the objective function (1) is

$$\begin{aligned} \sum_{j=1}^m \Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j)) &= \sum_{j=1}^m \sum_{l=1}^k ||\tilde{t}_l - c^j_{\delta_j(l)}||^2 \\ &= \sum_{l=1}^k (\sum_{j=1}^m ||\tilde{t}_l - c^j_{\delta_j(l)}||^2). \end{aligned} \quad (4)$$

Since $\tilde{\mathcal{T}}$ is the optimal k-tuple minimizing the objective function, from (4) we know that each $\tilde{t}_l$ should be the mean point of $\{c^j_{\delta_j(l)} \mid 1 \le j \le m\}$. □

Proof. (**of Lemma 2**) Since each $c^j_{\delta_j(l)}$ is a boolean vector, we know that $\tilde{t}_l$ has non-negative value in each coordinate from Lemma 3. Furthermore,

$$\begin{aligned} \sum_{l=1}^k \tilde{t}_l &= \sum_{l=1}^k (\frac{1}{m} \sum_{j=1}^m c^j_{\delta_j(l)}) \\ &= \frac{1}{m} \sum_{j=1}^m \sum_{l=1}^k c^j_{\delta_j(l)} = \frac{1}{m} \sum_{j=1}^m \sum_{l=1}^k c^j_l. \end{aligned} \quad (5)$$

Note that $\sum_{l=1}^k c^j_l = \underbrace{(1, \cdots, 1)}_{N}$. Thus $\sum_{l=1}^k \tilde{t}_l = \underbrace{(1, \cdots, 1)}_{N}$ as well. □

Relaxation to fractional memberships. From Lemma 2, we know that the k values in each i-th coordinate from $\tilde{\mathcal{T}}$ are non-negative with a summation equal to 1. This can be interpreted as they forming a probabilistic distribution. In other words, for each $a_i \in \mathcal{A}$, we can claim its membership to k clusters based on this distribution. Although this is not an exact solution for ensemble clustering, it is acceptable and makes sense in practice. For instance, we may claim that one object belongs to class 1, 2, and 3 with probabilities of 70%, 20%, and 10%, respectively. Moreover, each minimum weight bipartite matching $\Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j))$ can be viewed as the clustering difference between a probabilistic clustering result $\tilde{\mathcal{T}}$ and a hard clustering result $p(\mathcal{C}_j)$.

4. THE ALGORITHM

4.1 The Basic Algorithm

We first present our algorithm for basic ensemble clustering, and then show how to accommodate the algorithm in a distributed setting in Section 4.2.

Key observation: From Lemma 3, we know that the points $\bigcup_{j=1}^{m} p(\mathcal{C}_j)$ associated with the matchings $\{\delta_j \mid 1 \leq j \leq m\}$ actually form k clusters, *i.e.,* $\{c^j_{\delta_j(l)} \mid 1 \leq j \leq m\}$ for $l = 1, \cdots, k$, and each $\tilde{t}_l$ serves as the cluster center, which is similar to the well known k-means but has the particular "bipartite matching" requirement from Definition 2, *i.e.,* each cluster has one and only one point from each $p(\mathcal{C}_j)$. The most common and easily-implementable algorithm for k-means is *Lloyd's algorithm* [29], which randomly selects k initial cluster centers, and then alternatively updates the partition and cluster centers iteratively. Although the objective value decreases after each iteration, there is no any strict quality guarantee in theory. However, we find that a similar strategy for ensemble clustering actually ensures a quality guaranteed solution based on some new geometric insights.

Lemma 4. *Given an instance of ensemble clustering as in Definition 1, one can randomly select a k-tuple points from $\{p(\mathcal{C}_j) \mid 1 \leq j \leq m\}$ as the initial solution, such that with probability $1/2$, it yields a 6-approximation with respect to the objective function (1).*

Proof. First, we know that the objective value (1) is $\sum_{j=1}^{m} \Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j))$, which is the sum of the costs caused by $\{p(\mathcal{C}_j) \mid 1 \leq j \leq m\}$. So if one randomly select one k-tuple, denoted as $p(\mathcal{C}_{j_0})$ *w.l.o.g.*, then from Markov inequality we know that its contribution to the objective value is no more than 2 times of the average cost with probability at least $1/2$, *i.e.,*

$$\begin{aligned}\Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_{j_0})) &\leq 2\frac{1}{m}\sum_{j=1}^{m}\Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j)) \\ &= 2\frac{1}{m}\sum_{j=1}^{m}\sum_{l=1}^{k}||\tilde{t}_l - c^j_{\delta_j(l)}||^2. \qquad (6)\end{aligned}$$

Furthermore, if we replace $\tilde{\mathcal{T}}$ by $p(\mathcal{C}_{j_0})$ as an initial solution, the objective value becomes

$$\begin{aligned}&\sum_{j=1}^{m}\sum_{l=1}^{k}||c^{j_0}_{\delta_{j_0}(l)} - c^j_{\delta_j(l)}||^2 \qquad (7)\\ \leq\ &\sum_{j=1}^{m}\sum_{l=1}^{k}(||c^{j_0}_{\delta_{j_0}(l)} - \tilde{t}_l|| + ||\tilde{t}_l - c^j_{\delta_j(l)}||)^2 \qquad (8)\\ \leq\ &\sum_{j=1}^{m}\sum_{l=1}^{k}(2||c^{j_0}_{\delta_{j_0}(l)} - \tilde{t}_l||^2 + 2||\tilde{t}_l - c^j_{\delta_j(l)}||^2) \qquad (9)\\ =\ &2m\sum_{l=1}^{k}||c^{j_0}_{\delta_{j_0}(l)} - \tilde{t}_l||^2 + 2\sum_{j=1}^{m}\sum_{l=1}^{k}||\tilde{t}_l - c^j_{\delta_j(l)}||^2 (10)\\ \leq\ &6\sum_{j=1}^{m}\sum_{l=1}^{k}||\tilde{t}_l - c^j_{\delta_j(l)}||^2 = 6\sum_{j=1}^{m}\Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j)), \qquad (11)\end{aligned}$$

where (8) and (9) follow from triangle inequality and the fact that $(a+b)^2 \leq 2a^2 + 2b^2$ for any a and $b \in \mathbb{R}$, and (11) follows from (6). Finally, (11) implies that $p(\mathcal{C}_{j_0})$ yields a 6-approximation. □

```
Algorithm 1 Basic Algorithm
Input: $\{p(\mathcal{C}_j) \mid 1 \leq j \leq m\}$, $k \in \mathbb{Z}^+$, and $\eta \in (0,1)$.
Output: An approximate ensemble clustering solution.
  1. Randomly select $\lceil \log \frac{1}{\eta} \rceil$ $k$-tuples from $\{p(\mathcal{C}_j) \mid 1 \leq j \leq m\}$, and choose the one with the smallest objective value (denoted as $\mathcal{T} = \{t_1, \cdots, t_k\}$) as the initial solution.
  2. Iteratively perform the following steps until the objective value becomes stable.
     (a) Update the minimum weight bipartite matching $\delta_j$ from $p(\mathcal{C}_j)$ to $\mathcal{T}$ for each $1 \leq j \leq m$ using Hungarian algorithm.
     (b) Update $t_l$ to be the mean of $\{c^j_{\delta_j(l)} \mid 1 \leq j \leq m\}$ for each $1 \leq l \leq k$.
```

Boost the success probability. In fact, we can select multiple k-tuples to increase the success probability in Lemma 4. Given any small $\eta \in (0,1)$, we can select $\log \frac{1}{\eta}$ k-tuples and choose the one with the smallest objective value; then the success probability becomes

$$1 - (1/2)^{\log \frac{1}{\eta}} = 1 - \eta. \qquad (12)$$

For example, if the probability is required to be 90%, we just need to select $\lceil \log 10 \rceil = 4$ k-tuples.

Now, we are ready to introduce our Algorithm 1 for solving ensemble clustering. Roughly speaking, Step 1 provides a 6-approximation as the initial solution according to Lemma 4, and Step 2 yields a local improvement. In both Step 1 and 2(a), for each $1 \leq j \leq m$ building the bipartite graph takes $O(k^2 N)$ time and computing the matching δ_j via Hungarian algorithm costs $O(k^3)$ time. Note that usually $k \ll N$, thus the total complexity for obtaining each δ_j is $O(k^2 N)$. In Step 2(b), similar to Lemma 3, we know that each t_l should be the mean point of $\{c^j_{\delta_j(l)} \mid 1 \leq j \leq m\}$ in order to reduce the objective value. Thus, the algorithm improves the objective value in each round of Step 2. From the above analysis we have the following theorem.

Theorem 2. *Algorithm 1 outputs a 6-approximation with probability $1 - \eta$, and the objective value converges in each round of Step 2. The total time complexity is $O((\log \frac{1}{\eta} + s)mk^2 N)$, where s is the number of rounds performed in Step 2.*

Note that the value of s does not influence the approximation ratio "6", however, in practice (such as our experiment) the objective value reduces significantly in the first 3-5 rounds. So usually we set $s \leq 5$. Moreover, as what we will show in Section 4.3, the influence of s is quite limited in the communication complexity after applying dimension reduction.

4.2 Adapting to Distributed Settings

Now, we consider the distributed settings described in Section 2.1 and show how to adapt Algorithm 1 to it. We will first discuss the two steps of Algorithm 1 separately, and then analyze the communication complexity.

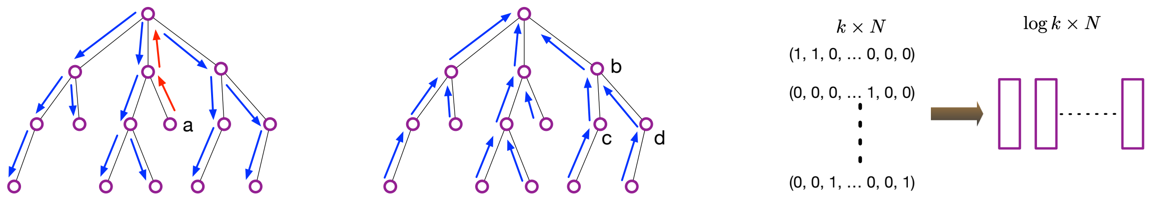


Figure 4: (a) and (b) illustrate the procedure of performing Step 1 on tree $\mathbb{H}$: node a is randomly selected, and it transmits its stored k-tuple to the root firstly; then the root broadcasts this k-tuple to all the nodes; the total cost is aggregated from the leaves, *e.g.,* node b needs to sum its own value of Δ and the values transmitted from its children c and d. (c) illustrates how to encode the k-tuple $p(\mathcal{C}_{j_0})$ into $\log k \times N$ bits, where the left is a $k \times N$ boolean matrix representing the k points of $p(\mathcal{C}_{j_0})$ in $\mathbb{R}^N$ (there is one and only one entry equaling to 1 in each column) and in the right each column of the $\log k$ bits represents one column in the left.

Step 1. Each time for randomly selecting a k-tuple, the root can randomly determine a path to some node in $\mathbb{H}$, then the node transmits its stored k-tuple, say $p(\mathcal{C}_{j_0})$, to the root through this path, and the root can broadcast $p(\mathcal{C}_{j_0})$ to all the nodes in $\mathbb{H}$. When each node obtains the selected $p(\mathcal{C}_{j_0})$, it computes the function Δ with its own k-tuple. Then starting from the leaves of $\mathbb{H}$, each node sums its own value of Δ and the values transmitted from its children, and returns this summation to its parent. Finally, the root has the total objective value $\sum_{j=1}^{m} \Delta(p(\mathcal{C}_{j_0}), p(\mathcal{C}_j))$. See Fig. 4(a) and (b) for an illustration. To complete Step 1, we can repeat this procedure $\log\frac{1}{\eta}$ times, and broadcast the final determined k-tuple with the smallest objective value to all nodes.

Step 2. In each round of Step 2, it is also a bottom-to-top procedure similar to the strategy for computing total cost $\sum_{j=1}^{m} \Delta(p(\mathcal{C}_{j_0}), p(\mathcal{C}_j))$ in Step 1. Once the root broadcasts $\mathcal{T}$, each node computes its bipartite matching δ_j. Note that each point $t_l \in \mathcal{T}$ should be updated to the mean point of $\{c^j_{\delta_j(l)} \mid 1 \le j \le m\}$ according to Step 2(b), *i.e.,*

$$t_l \Longrightarrow \frac{1}{m}\sum_{j=1}^{m} c^j_{\delta_j(l)}. \tag{13}$$

We can first ignore the front $\frac{1}{m}$ temporarily, and only aggregate $\sum_{j=1}^{m} c^j_{\delta_j(l)}$. For each $1 \le l \le k$, each node sums $c^j_{\delta_j(l)}$ to the corresponding points (with respect to the index l) from its children; then the node transmits these k new points after the summation to its parent. Finally, the root obtains the point-set $\{\sum_{j=1}^{m} c^j_{\delta_j(l)} \mid 1 \le l \le k\}$, and just needs to average each of the points by m. Thus, $\mathcal{T}$ is updated within this round.

Total communication cost. Now, we analyze the total communication cost of the two steps. Firstly, we note that each k-tuple $p(\mathcal{C}_{j_0})$ can be encoded by using $\log k \times N$ bits as follows. Since each point of $p(\mathcal{C}_{j_0})$ is a boolean vector and for each of the N coordinates there is one and only one of the k points having the value of 1 according to the construction in (2), $p(\mathcal{C}_{j_0})$ can be represented by a $\log k \times N$-bit matrix, where the i-th column indicates the point having value 1 in its i-th coordinate (see Fig. 4(c)). Since there are $m-1$ edges in $\mathbb{H}$, it results in $O(\log k \times Nm)$ communication cost for selecting and broadcasting one k-tuple in Step 1. Also from the definition of Δ and Lemma 1, we know that $\sum_{j=1}^{m} \Delta(p(\mathcal{C}_{j_0}), p(\mathcal{C}_j))$ is a non-negative integer no more than $2Nm$, which implies that each node just needs to transmit $O(\log(Nm))$ bits when aggregating the total cost $\sum_{j=1}^{m} \Delta(p(\mathcal{C}_{j_0}), p(\mathcal{C}_j))$. Consequently, Step 1 costs

$$O(\log\frac{1}{\eta}(\log k \times N + \log(Nm))m) = O(\log\frac{1}{\eta}\log(km) \times Nm) \tag{14}$$

total communication cost.

The communication cost of Step 2 follows from a similar analysis, where the only difference is that the cost for transmitting $\{\sum_{j=1}^{m} c^j_{\delta_j(l)} \mid 1 \le l \le k\}$ (or $\mathcal{T}$) is no longer $\log k \times N$. For convenience, we just consider $\{\sum_{j=1}^{m} c^j_{\delta_j(l)} \mid 1 \le l \le k\}$ since $\mathcal{T}$ can be easily obtained by multiplying the factor of $\frac{1}{m}$. The k points in $\{\sum_{j=1}^{m} c^j_{\delta_j(l)} \mid 1 \le l \le k\}$ can be similarly represented as a $k \times N$ matrix, with each entry being an integer between 0 and m. Thus the $k \times N$ matrix can be encoded by using $k \times N \times \log m$ bits. If the algorithm runs s rounds in Step 2, the total communication cost is

$$O(skNm\log m). \tag{15}$$

In total, the communication complexity of the whole Algorithm 1 is

$$O((\log\frac{1}{\eta}\log(km) + sk\log m)Nm). \tag{16}$$

From the above analysis, we can see two advantages of our algorithm. (1) The communication complexity is alway nearly linear in m and **independent of the topology of** $\mathbb{H}$; (2) it is a solution without any sacrifice in quality comparing to the aggregated solution in [34]. Thus, we have following theorem.

THEOREM 3. *Algorithm 1 can be implemented without sacrificing any quality of solution in a distributed setting where all the nodes form a communication network containing a spanning tree $\mathbb{H}$ of arbitrary shape, and the total communication complexity is $O((\log\frac{1}{\eta}\log(km) + sk\log m)Nm)$.*

4.3 Reducing Communication Cost via Dimension Reduction

From Section 4.2 we know that the communication complexity is linear in the dimensionality N, which is also the number of observed objects in Definition 1 and can be very large in practice. Thus, a smaller N can result in a significant reduction on the communication complexity.

The famous *Johnson-Lindenstrauss (JL)*-Lemma [26] tells us that if there are n points in any dimensional Euclidean space, one can randomly project all the points to an $O(\frac{\log n}{\epsilon^2})$-dimensional sub-space and approximately preserves all the pairwise distances for any given small parameter $\epsilon > 0$. Note

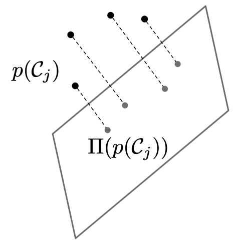


Figure 5: Each k-tuple $p(\mathcal{C}_j)$ is projected to the subspace, and the projection is $\Pi(p(\mathcal{C}_j))$.

that preserving pairwise distances does not immediately imply a quality-guaranteed solution for our ensemble clustering problem, since the feasible domain of the solution (*i.e.,* the k-tuple $\mathcal{T}$) is the whole continual space $\mathbb{R}^N$; this means that there are infinite number of possible k-tuples as the candidates for $\mathcal{T}$, but JL-Lemma can only preserve a finite number of pairwise distances. The following lemma resolves this issue and gives a positive answer to the problem.

LEMMA 5. *Let $d = O(\frac{\log(km)}{\epsilon^2})$, $\epsilon \in (0,1)$, and Π be a random projection $\mathbb{R}^N \longmapsto \mathbb{R}^d$ for $\{p(\mathcal{C}_j) \mid 1 \le j \le m\}$. Then with high probability, any α-approximation (for any $\alpha > 1$) of the ensemble clustering problem in $\mathbb{R}^d$ yields an $\alpha(\frac{1+\epsilon}{1-\epsilon})^2$-approximation in the original $\mathbb{R}^N$.*

Prior proving Lemma 5, we need the following basic result (see Appendix for the proof).

LEMMA 6. *Given a set of points $\mathcal{X}$ in any Euclidean space, $\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{X}}||x-y||^2 = 2|\mathcal{X}|\sum_{x\in\mathcal{X}}||x-o||^2$, where o is the mean point of $\mathcal{X}$.*

Lemma 6 reveals the relationship between the total pairwise squared distances and the variance.

PROOF. (**of Lemma 5**) Note that there are $|\bigcup_{j=1}^m p(\mathcal{C}_j)| = km$ points in total. From JL-Lemma, we know that with high probability Π preserves the pairwise distances, *i.e.,* for any two points x, $y \in \bigcup_{j=1}^m p(\mathcal{C}_j)$,

$$\begin{aligned}(1-\epsilon)\frac{d}{N}||x-y||^2 &\le ||\Pi(x)-\Pi(y)||^2 \\ &\le (1+\epsilon)\frac{d}{N}||x-y||^2. \quad (17)\end{aligned}$$

Suppose that $\mathcal{T}_\Pi = \{t_{\Pi,l} \mid 1 \le l \le k\}$ and $\tilde{\mathcal{T}}_\Pi = \{\tilde{t}_{\Pi,l} \mid 1 \le l \le k\}$ are respectively an α-approximation and an optimal solution of the ensemble clustering for the set of ponts $\{\Pi(p(\mathcal{C}_j)) \mid 1 \le j \le m\}$ in $\mathbb{R}^d$. Then,

$$\sum_{j=1}^m \Delta(\mathcal{T}_\Pi, \Pi(p(\mathcal{C}_j))) \le \alpha \sum_{j=1}^m \Delta(\tilde{\mathcal{T}}_\Pi, \Pi(p(\mathcal{C}_j))). \quad (18)$$

We also denote the corresponding bipartite matching determined by $\Delta(\mathcal{T}_\Pi, \Pi(p(\mathcal{C}_j)))$ and $\Delta(\tilde{\mathcal{T}}_\Pi, \Pi(p(\mathcal{C}_j)))$ as π_j and $\tilde{\pi}_j$ respectively for each j. Now, we construct a mapping f which transforms $\mathcal{T}_\Pi$ and $\tilde{\mathcal{T}}_\Pi$ to the solutions of ensemble clustering on $\{p(\mathcal{C}_j) \mid 1 \le j \le m\}$ in the original space $\mathbb{R}^N$.

$$f(t_{\Pi,l}) = \frac{1}{m}\sum_{j=1}^m c^j_{\pi_j(l)},\ f(\mathcal{T}_\Pi) = \{f(t_{\Pi,l}) \mid 1 \le l \le k\}; \quad (19)$$

$$f(\tilde{t}_{\Pi,l}) = \frac{1}{m}\sum_{j=1}^m c^j_{\tilde{\pi}_j(l)},\ f(\tilde{\mathcal{T}}_\Pi) = \{f(\tilde{t}_{\Pi,l}) \mid 1 \le l \le k\}. \quad (20)$$

Actually, the above construction of f is quite intuitive. According to Lemma 3, each $t_{\Pi,l}$ and $\tilde{t}_{\Pi,l}$ are the means of the corresponding clusters in $\mathbb{R}^d$; then $f(t_{\Pi,l})$ and $f(\tilde{t}_{\Pi,l})$ are still their means in $\mathbb{R}^N$. Additionally, since Π is a linear mapping, for each $1 \le l \le k$ we have

$$\Pi(f(t_{\Pi,l})) = t_{\Pi,l} \textit{ and } \Pi(f(\tilde{t}_{\Pi,l})) = \tilde{t}_{\Pi,l}. \quad (21)$$

In the remaining of the proof, we will show that $f(\mathcal{T}_\Pi)$ is an $\alpha(\frac{1+\epsilon}{1-\epsilon})^2$-approximation. From Lemma 6, (17), (18), and (21), we know that

$$\begin{aligned}\sum_{j=1}^m \Delta(f(\mathcal{T}_\Pi), p(\mathcal{C}_j)) &\le \frac{1}{1-\epsilon}\frac{N}{d}\sum_{j=1}^m \Delta(\mathcal{T}_\Pi, \Pi(p(\mathcal{C}_j))) \quad (22)\\ &\le \frac{1}{1-\epsilon}\frac{N}{d}\alpha\sum_{j=1}^m \Delta(\tilde{\mathcal{T}}_\Pi, \Pi(p(\mathcal{C}_j))) \quad (23)\\ &\le \frac{1+\epsilon}{1-\epsilon}\alpha\sum_{j=1}^m \Delta(f(\tilde{\mathcal{T}}_\Pi), p(\mathcal{C}_j)). \quad (24)\end{aligned}$$

Recall that $\tilde{\mathcal{T}}$ is the optimal solution in the original $\mathbb{R}^d$. By a similar strategy in (22)-(24), we have

$$\sum_{j=1}^m \Delta(f(\tilde{\mathcal{T}}_\Pi), p(\mathcal{C}_j)) \le \frac{1+\epsilon}{1-\epsilon}\sum_{j=1}^m \Delta(\tilde{\mathcal{T}}, p(\mathcal{C}_j)). \quad (25)$$

Combining (24) and (25), we complete the proof. □

From Lemma 5 we know that through random projection, the dimensionality can be totally independent of the dimensionality N and the quality can still be guaranteed. Below we discuss how to realize such a dimension reduction in wireless sensor networks.

Realization. A nice property of the JL-Lemma is that the random projection matrix can be generated in multiple machines as long as they all use the same *random seed*; in other words, each node of $\mathbb{H}$ can independently generate a unified random $O(\frac{\log(km)}{\epsilon^2})$-dimensional subspace in advance. Then each k-tuple points is projected into $\mathbb{R}^{O(\frac{\log(km)}{\epsilon^2})}$, and Algorithm 1 is performed in this lower dimensional sub-space. Finally, we map the solution from $\mathbb{R}^{O(\frac{\log(km)}{\epsilon^2})}$ to $\mathbb{R}^N$ using a similar idea in (19) and (20), *i.e.,* computing the mean point of each cluster in $\mathbb{R}^N$ determined by the bipartite matchings. Note that this mapping can be similarly implemented in $\mathbb{H}$ as Step 2 in Section 4.2.

As for the communication complexity, we need several modifications compared to Theorem 3. As mentioned in Section 4.2, when transmitting $p(\mathcal{C}_{j_0})$ in our original Algorithm 1, it needs only $\log k \times N$ bits due to the nice property of the boolean vectors (Fig. 4(c)). However, the points of $\Pi(p(\mathcal{C}_{j_0}))$ are no longer boolean vectors. Instead, using the random sign matrix as random projection matrix [1], each point of $\Pi(p(\mathcal{C}_{j_0}))$ becomes an $O(\frac{\log(km)}{\epsilon^2})$-dimensional vector where the value in each dimension is an integer between $-N$ and N. Consequently, the communication complexity of Algorithm 1 after dimension reduction becomes

$$\log\frac{1}{\eta}k\frac{\log(km)}{\epsilon^2}(\log N)m + sk\log(Nm)\frac{\log km}{\epsilon^2}m, \quad (26)$$

where the first term is for Step 1 and the second term is for Step 2 of Algorithm 1. With simple calculations, we know that (26) equals to

$$O(k\frac{\log(km)}{\epsilon^2}m(\log\frac{1}{\eta}\log N + s\log(Nm)))$$
$$= O(\frac{\log(km)}{\epsilon^2}km(\log\frac{1}{\eta}+s)\log(Nm)). \quad (27)$$

Finally, in the mapping step the k-tuple $f(\mathcal{T}_\Pi)$ (recall the proof of Lemma 5) is aggregated in $\mathbb{H}$ which costs

$$O(kNm\log m) \quad (28)$$

communication cost (similar to (15) but without the factor s).

In summary, we have the following theorem.

Theorem 4. *If all the points $\bigcup_{j=1}^{m} p(\mathcal{C}_j)$ are randomly projected to a subspace $\mathbb{R}^{O(\frac{\log km}{\epsilon^2})}$, the approximation ratio becomes $6(\frac{1+\epsilon}{1-\epsilon})^2$, and the communication complexity is reduced to*

$$O(\frac{\log(km)}{\epsilon^2}km(\log\frac{1}{\eta}+s)\log(Nm)+kNm\log m). \quad (29)$$

Remark 3. *Actually, when N is very large, the second term of (29) dominates the whole communication complexity. In other words, (29) can be written as $O(kNm\log m)$ in short. Compared to the communication complexity in Theorem 3 which is $O((\log\frac{1}{\eta}\log(km)+sk\log m)Nm)$, the reduction is quite significant.*

Difference with compressed sensing based data gathering. It is worthy pointing out that our JL-lemma based distributed ensemble clustering is significantly different from those compressed sensing based data gathering techniques. Data gathering in a sensor network is to collect data d_i from each sensor i through the wireless network formed by the sensors. To reduce the communication cost, compressed sensing [9] has been widely used in data gathering [30, 37, 23]. Note that although compressed sensing is closely related to JL-Lemma in mathematics [6], these compressed sensing based approaches are in general not applicable to our distributed ensemble clustering problem. There are several reasons. Firstly, compressed sensing requires the values acquired by the sensors to form a sparse vector (or have a sparse representation); but the data $\{p(\mathcal{C}_j) \mid 1 \leq j \leq m\}$ in our problem does not necessarily have such a property. Secondly, according to our algorithmic framework in Section 4.1 and 4.2 it is not necessary to know each $p(\mathcal{C}_j)$, instead, we just need to aggregate their summations based on each individual δ_j. Roughly speaking, compressed sensing for data gathering is to compress the whole data-set in a global manner, while our approach is to use JL-lemma to locally compress the data in each individual sensor (node).

5. PERFORMANCE EVALUATION

In order to evaluate the performance, we test our algorithm on two benchmark datasets and present below the clustering errors and communication costs.

Datasets. We use two benchmark datasets from *UCI machine learning repository* for clustering purpose: **(1).** *Plants* from USDA Plants Database and **(2).** *US Census* from the U.S. Department of Commerce Bureau, which contain $N = 22632$ and $N = 238713$ objects respectively. For each

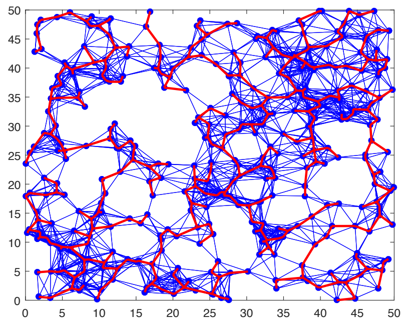


Figure 6: Network topology

dataset, we run k-means to obtain a clustering result as the ground truth (where k is set to be 10), and then generate $m = 100$ to 400 different clustering solutions based on a given noise level. For example, if the noise level is $e \in (0, 1)$, we generate each of the m clustering solutions as follows: randomly pick $e * N$ objects and change their cluster memberships in the ground truth; we also merge $0.1 * k$ pairs of clusters which are randomly picked from the ground truth.

Network topology. As illustrated in Fig. 6, m (= 100 to 400) nodes are firstly randomly placed on a 50×50 region, and their unit disk graph $\mathbb{G}$ is then built. According to the problem setting in Section 2.1, we require that $\mathbb{G}$ is connected (note that we can always double the radius of the disk until $\mathbb{G}$ becomes connected). Finally, a spanning tree $\mathbb{H}$ (shown in Fig. 6 by thick red lines) of $\mathbb{G}$ is computed, and the communication is restricted between only parents and their children.

Results. The computation in each node is simulated on a Linux workstation (2 GHz Intel Core i7 and 4 GB Meomory). For showing the advantage from dimension reduction, we also conduct the experiment after projecting all the data from $\mathbb{R}^N$ to a randomly selected subspace $\mathbb{R}^d$, where d is set to be $\lceil\frac{8}{0.3^2}\log km\rceil$, $\lceil\frac{6}{0.3^2}\log km\rceil$, and $\lceil\frac{4}{0.3^2}\log km\rceil$, respectively. For each noise level $e = \{0.25, 0.30, \cdots, 0.70\}$, the experiment is repeated 10 times and both of the average clustering error and communication cost are reported in Fig. 7.

For comparisons with existing approaches, we use Meta-CLustering Algorithm (MCLA) [32] to produce the baselines. Note that there are also a couple of other existing approaches (see [18]), but most of them either need to build a correlation graph with quadratic complexity or solve some complicated optimization problem (*e.g.,* semidefinite programming [33]), which could cost several hours or even run out of memory in a local workstation (for a setting of N between $2*10^4$ to $2*10^5$ and $m = 100$ to 400). We use two ways to simulate MCLA in distributed settings: (1) every node sends its data to the root, and the root performs MCLA on the global dataset; (2) each node performs MCLA on the data from its children and itself, reports the result to its parent, and finally the root obtains an aggregated solution.

Analysis. From Fig. 7, we can see that the clustering error does not change too much when the dimension is reduced, while the difference in communication cost is large, which implies that the benefit from dimension reduction is significant while the sacrifice in accuracy is neglectable. Furthermore, the difference in communication cost with respect to

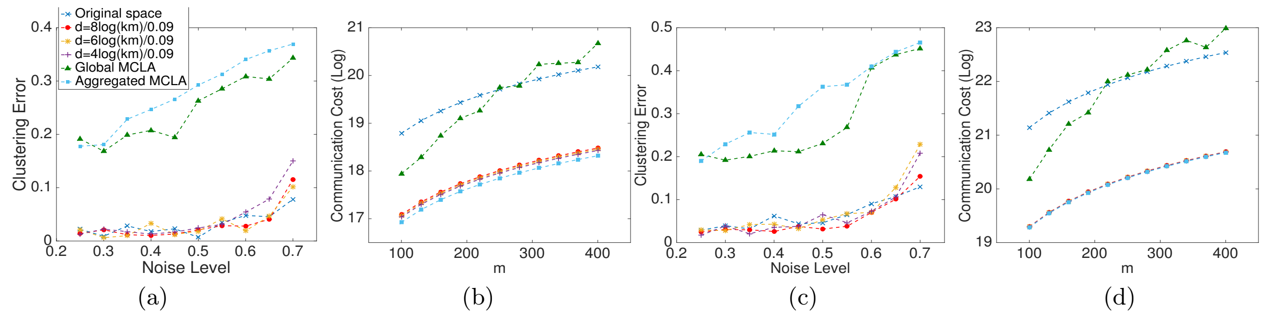


Figure 7: (a) and (b) are for Plants dataset; (c) and (d) are for US Census dataset.

different values of d is small, because due to Theorem 4, the communication complexity is dominated by the last term $kNm \log m$ especially when N is much larger than k and m. Comparing to baselines by global and aggregated MCLA, our clustering errors are much lower; aggregated MCLA has similar communication cost to ours with dimension reduction since it also needs to transmit the data along the tree once, which is roughly $kNm \log m$; global MCLA has much larger communication cost since it needs to transmit all the data to the root, and the communication complexity could be as worse as quadratic in m.

6. CONCLUSION

In this paper, we present an efficient distributed ensemble clustering framework for networked sensing systems where an object is usually observed by multiple sensors. Specifically, we first propose a novel formulation for ensemble clustering based on some new geometric insights, and then give an efficient algorithm with quality guarantee. We also show how to extend our approach to distributed settings, and reduce the communication complexity through dimension reduction. Experiments on benchmark datasets suggest that our approach outperforms baselines on both clustering accuracy and communication cost.

7. ACKNOWLEDGMENTS

The first and third authors were supported in part by NSF through grants CCF-1422324, IIS- 1422591, and CNS-1547167. The first author was also supported in part by a start-up fund from Michigan State University. Part of the work was done when the first author was in IIIS, Tsinghua University and Simons Institute, UC Berkeley. The second author was supported in part by the NSF through grant CNS-1566374.

8. REFERENCES

[1] D. Achlioptas, "Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 671-687, 2003.

[2] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: ranking and clustering," *STOC* 2005: 684-693.

[3] M. -F. Balcan, A. Blum, S. Fine, and Y. Mansour, "Distributed Learning, Communication Complexity and Privacy," *COLT* 2012: 26.1-26.22.

[4] M.-F. Balcan, A. Blum, and A. Gupta, "Clustering under Approximation Stability," *J. ACM* 60(2): 8 (2013).

[5] M.-F. Balcan, S. Ehrlich, and Y. Liang, "Distributed k-means and k-median clustering on general communication topologies," *NIPS* 2013: 1995-2003.

[6] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "The Johnson-Lindenstrauss Lemma Meets Compressed Sensing," *Preprint*, 2006.

[7] K. C. Barr and K. Asanovic, "Energy aware lossless data compression," *MobiSys* 2003.

[8] R. R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. of the IEEE* 91(8): 1163-1171 (2003).

[9] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*,25(2):21-30, Mar. 2008.

[10] X. Cheng, J. Xu, J. Pei, and J. Liu, "Hierarchical distributed data classification in wireless sensor networks," *MASS* 2009: 10-19.

[11] J. Chin, N. Rao, D. Yau, M. Shankar, Y. Yang, J. Hou, S. Srivathsan, and S. Iyengar, "Identification of low-level point radioactive sources using a sensor network," *TOSN* 7(3) (2010).

[12] T. Coleman and A. Wirth, "A Polynomial Time Approximation Scheme for k-Consensus Clustering," *SODA* 2010: 729-740.

[13] H. Daumé III, J. M. Phillips, A. Saha, and S. Venkatasubramanian, "Efficient Protocols for Distributed Classification and Optimization," *ALT* 2012: 154-168.

[14] H. Ding and J. Xu, "A Unified Framework for Clustering Constrained Data without Locality Property," *SODA'15*, pp. 1471-1490, 2015.

[15] D. Duran, D. Peng, H. Sharif, B. Chen, and D. Armstrong, "Hierarchical character oriented wildlife species recognition through heterogeneous wireless sensor networks," *PIMRC* 2007: 1-5.

[16] X. Z. Fern and C. E. Brodley, "Solving cluster ensemble problems by bipartite graph partitioning," *ICML* 2004: 36.

[17] J. Gao, F. Liang, W. Fan, Y. Sun, and J. Han, "Graph-based Consensus Maximization among Multiple Supervised and Unsupervised Models," *NIPS* 2009: 585-593.

[18] J. Ghosh and A. Acharya, "Cluster Ensembles: Theory and Applications," *Data Clustering: Algorithms and Applications* 2013: 551-570.

[19] L. Girod, M. Lukac, V. Trifa, and D. Estrin, "The design and implementation of a self-calibrating distributed acoustic sensing platform," *SenSys'06*, pp .71-84.

[20] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J. A. Stankovic, T. Abdelzaher, and B. H. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments," *SenSys* 2005: 205-217.

[21] Y. Guo, P. Corke, G. Poulton, T. Wark, G. Bishop-Hurley, and D. Swain, "Animal behaviour understanding using wireless sensor networks," *LCN* 2006: 607-614.

[22] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ICDE* 2005: 341-352.

[23] J. Haupt, W. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 92-101, 2008.

[24] J. Hill and D. Culler, "Mica: A Wireless Platform for Deeply Embedded Networks," *IEEE Micro* 22(6): 12-24 (2002).

[25] A. K. Jain, M. N. Murty, and P. J. Flynn, " Data Clustering: A Review," *ACM Comput. Surv. (CSUR)* 31(3): 264-323 (1999).

[26] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Conference in Modern Analysis and Probability (New Haven, Conn., 1982)*, Contemporary Mathematics 26, Providence, RI: American Mathematical Society, 189-06.

[27] X. Li and Y. Wang, "Wireless sensor networks and computational geometry," *Handbook of Sensor Networks, CRC Press, USA (2003).*

[28] Y. Liang, M. -F. Balcan, V. Kanchanapally, and D. P. Woodruff, "Improved Distributed Principal Component Analysis," *NIPS* 2014: 3113-3121.

[29] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory* 28 (2): 129-137 (1982).

[30] C. Luo, F. Wu, J. Sun, and C. Chen, "Compressive data gathering for large-scale wireless sensor networks," *MOBICOM*, pp. 145-156, 2009.

[31] R. Pon, M. Batalin, J. Gordon, A. Kansal, D. Liu, M. Rahimi, L. Shirachi, Y. Yu, M. Hansen, W. Kaiser, M. Srivastava, G. Sukhatme, and D. Estrin, "Networked infomechanical systems: a mobile embedded networked sensor platform," *IPSN*, pp. 376-381, 2005

[32] A. Strehl and J. Ghosh, "Cluster Ensembles - A Knowledge Reuse Framework for Combining Partitionings," *AAAI* 2002: 93-98.

[33] V. Singh, L. Mukherjee, J. Peng, and J. Xu, "Ensemble clustering using semidefinite programming with applications," *Machine Learning* 79(1-2): 177-200 (2010).

[34] L. Su, J. Gao, Y. Yang, T. F. Abdelzaher, B. Ding, and J. Han, "Hierarchical Aggregate Classification with Limited Supervision for Data Reduction in Wireless Sensor Networks," *SenSys* 2011: 40-53.

[35] L. Su, S. Hu, S. Li, F. Liang, J. Gao, T. F. Abdelzaher, and J. Han, "Quality of Information Based Data Selection and Transmission in Wireless Sensor Networks," *RTSS* 2012: 327-338.

[36] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," *Pervasive* 2004: 158-175.

[37] J. Wang, S. Tang, B. Yin, and X. Li, "Data gathering in wireless sensor networks through intelligent compressive sensing," *INFOCOM'12*, pp. 603-611, 2012.

[38] Y. Yang, L. Wang, D. Noh, H. Le, and T. Abdelzaher, "SolarStore: enhancing data reliability in solar-powered storage-centric sensor networks," *MobiSys'09*, pp. 333-346.

[39] B. Yu, J. Li, and Y. Li, "Distributed Data Aggregation Scheduling in Wireless Sensor Networks," *INFOCOM'09*, pp. 2159-2167.

[40] Z. Zeng, S. Yu, W. Shin, and J. C. Hou, "PAS: A Wireless-Enabled, Cell-Phone-Incorporated Personal Assistant System for Independent and Assisted Living," *ICDCS* 2008: 233-242.

APPENDIX

A. CLUSTERING ERROR FUNCTION Δ

Suppose that $\mathcal{C} = \{S_1, \cdots, S_k\}$ is the ground truth clustering for the set of objects $\mathcal{A}$, and $\mathcal{C}' = \{S'_1, \cdots, S'_k\}$ is another clustering result. We define the clustering error made by $\mathcal{C}'$ as the total number of wrongly clustered objects over $|\mathcal{A}|$, *i.e.*,

$$Err(\mathcal{C}') = \min_{\delta} \frac{1}{|\mathcal{A}|} \sum_{j=1}^{k} |S'_j \setminus S_{\delta(j)}|, \tag{30}$$

where δ is any bijection from $[k]$ to $[k]$.

Claim. *For any δ, $\sum_{j=1}^k |S'_j \setminus S_{\delta(j)}| = \frac{1}{2}\sum_{j=1}^k(|S'_j \setminus S_{\delta(j)}| + |S_{\delta(j)} \setminus S'_j|)$.*

Proof. Since each object from $\mathcal{A}$ appears in one and only one cluster of $\mathcal{C}$ and $\mathcal{C}'$, for any $1 \le j_1 \ne j_2 \le k$,

$$(S'_{j_1} \setminus S_{\delta(j_1)}) \cap (S'_{j_2} \setminus S_{\delta(j_2)}) = \emptyset \tag{31}$$

$$(S_{\delta(j_1)} \setminus S'_{j_1}) \cap (S_{\delta(j_2)} \setminus S'_{j_2}) = \emptyset. \tag{32}$$

Thus $\sum_{j=1}^k |S'_j \setminus S_{\delta(j)}| = |\cup_{j=1}^k S'_j \setminus S_{\delta(j)}|$ and $\sum_{j=1}^k(|S'_j \setminus S_{\delta(j)}| + |S_{\delta(j)} \setminus S'_j|) = |\cup_{j=1}^k S'_j \setminus S_{\delta(j)}| + |\cup_{j=1}^k S_{\delta(j)} \setminus S'_j|$. Also, for any $a \in S'_j \setminus S_{\delta(j)}$, there must exist some $j' \ne j$, such that $a \in S_{\delta(j')} \setminus S'_{j'}$, and vice versa. This means that $\cup_{j=1}^k S'_j \setminus S_{\delta(j)} = \cup_{j=1}^k S_{\delta(j)} \setminus S'_j$. Hence, $\sum_{j=1}^k |S'_j \setminus S_{\delta(j)}| = \frac{1}{2}\sum_{j=1}^k(|S'_j \setminus S_{\delta(j)}| + |S_{\delta(j)} \setminus S'_j|)$. The proof is completed. □

According to the definition for Δ and the above claim, we have $\Delta(\mathcal{C}, \mathcal{C}') = 2|\mathcal{A}|Err(\mathcal{C}')$.

B. PROOF OF LEMMA 6

Lemma 6 reveals the relationship between the total pairwise squared distances and the variance, and can be easily obtained by the following calculation.

$$\begin{aligned}
& \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{X}} ||x-y||^2 \\
= & \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{X}} ||x-o+o-y||^2 \\
= & \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{X}} (||x-o||^2 + 2 < x-o, o-y > + ||o-y||^2) \\
= & |\mathcal{X}| \sum_{x\in\mathcal{X}} ||x-o||^2 + |\mathcal{X}| \sum_{y\in\mathcal{X}} ||y-o||^2 \\
= & 2|\mathcal{X}| \sum_{x\in\mathcal{X}} ||x-o||^2,
\end{aligned} \tag{33}$$

where $< x-o, o-y >$ denotes the inner product of $x-o$ and $o-y$, and the third equality follows from the fact $\sum_{y\in\mathcal{X}} < x-o, o-y > = < x-o, \sum_{y\in\mathcal{X}}(o-y) > = 0$.