IE 529 Project Report

Distributed k-Means and k-Median Clustering on General Topologies

by Maria Florina Balcan, Steven Ehrlich and Yingyu Liang

Introduction:

In the era of Big Data, it is becoming more and more difficult to store data on a single memory. Datasets are being stored on distributed systems such as distributed databases and networks. Algorithms that work on distributed systems have become the need of the hour. This paper tackles the problem of clustering data on distributed datasets, specifically the k-mean and k-medians clustering. The paper introduces novel distributed clustering algorithms for k-means and k-median where the data is stored over multiple locations (nodes) and communication is restricted to the edges in the graph, with improvements in communication complexity while preserving provable guarantees on the clustering quality. The algorithm manages to keep low communication costs by communicating only the sum of the costs of approximations to the local optimal clustering on each node to all other nodes. Thus only a single value is communicated from each node to all other nodes. The algorithm then uses these costs to construct local coresets.

Methodology:

The dataset is distributed on several nodes in an undirected connected graph. A local constant approximation solution is computed on each node and then the cost of each data point with it's respective local solution is calculated. The total cost of each local solution is then communicated to the other nodes in the graph using a message passing algorithm. Based on the costs of all nodes and the contribution of the data towards their local solution, the local data are sampled to form a local coreset which is a set of points, together with a weight for each point, such that the cost of this weighted set approximates the cost of the original data for any set of k centers.

Coreset Construction:

One of the major goals of the analysis is to construct distributed coresets for k-means and k-medians algorithm.

Intuition:

The points close to the nearest centers can approximately be well represented by the nearest centers but the points far away cannot. Hence, to design the coreset, a constant approximation solution for the whole dataset is created and then points proportional to their contribution to the cost of the solution are sampled.
Thus, a local approximation solution for each local dataset is created and the total costs of these local solutions are communicated. Points are then sampled proportional to their contribution to the cost of their local solutions. This creates distributed coresets over the nodes consisting of sampled points and centers in the local solutions.
Once the local coresets are constructed, the message passing algorithm is called once again to combine the local coresets. If each node constructs an ϵ-coreset on its local data, then the union of these local coresets is an ϵ-coreset for the entire data. The constant approximation algorithm is applied to this global coreset to obtain the final result.

Comparison to Other Coreset Algorithms:

This paper is compared to other coreset utilizing algorithms like Zhang et. al. [1] and [2]. Zhang et. al. is limited to rooted trees. Each node constructs a local coreset and passes it to its parent node. The parent node then combines the incoming coresets and constructs a coreset on the combination of its child coresets. This process continues until the root of the tree is reached. Thus the communication cost to develop coresets is very high as local coresets are communicated over the tree. This is very large compared to the algorithm developed by the authors where only the cost of each node (a single value) is communicated across the graph to construct the distributed coresets. The accuracy a coreset needs (and thus the overall communication complexity) scales with the height of the tree. Although it is possible to find a spanning tree in any communication network, when the graph has large diameter every tree has large height which greatly increases the size of coresets which must be communicated across the lower levels of the tree. [2] ignores the communication cost and merges coresets in a parallel computation model.

Preliminaries:

- Consider two points p, q $\in R^d$
  d(p, q) is the Euclidean distance between p and q
- K-means goal: Find set of k centers X = { x_1, x_2, ..., x_k } which minimize the k-means cost of data P $\subseteq R^d$
- K-means cost = Cost(P, X) = $\sum_{p \in P} d(p, X)^2$
  where $d(p, X) = \min d(p, x); \quad \forall x \in X$
- If P is a weighted dataset with a weight function w, then
  K-means cost = Cost(P, X) = $\sum_{p \in P} w(p)\, d(p, X)^2$
- For distributed clustering, we consider
  Set of n nodes V = { v_i, 1 ≤ i ≤ n } which communicate on undirected connected graph
  G = (V, E), with m = $|E|$ edges
- (v_i, v_j) $\in$ E indicates that v_i and v_j can communicate with each other.
- Communication cost = number of points transmitted. For simplicity, assume that there is no latency in communication.
- Each node v_i contains local set of data points P_i and the global dataset is P = $\bigcup_{i=1}^{n} P_i$

Technical Details:

The following definitions are important for the details of the proofs that follow:

Definition 1: (*Coreset*)

An ϵ-coreset for a set of points P with respect to a center-based cost function is a set of points S and a set of weights $w : S \rightarrow R$ such that for any set of centers X,

$$(1-\epsilon)cost(P,X) \leq \sum_{p \in S} w(p)cost(p,X) \leq (1+\epsilon)cost(P,X)$$

Definition 2: (*Dimension of the function space*)

Let F be a finite set of functions from a set P to $R_{\geq 0}$. Then for $f \in F$, define the set $B(f, r) = \{p : f(p) \leq r\}$. Then the dimension of the function space D(F, P) is the smallest integer d such that for any G $\subseteq P$, $|\{G \cap B(f, r) : f \in F,\ r \geq 0\}| \leq |G|^d$.

The objective is to develop an algorithm that can compute coresets for the data in a distributed fashion, while maintaining a desirable cost approximation ratio. The following theorem serves as the backbone for development of the algorithm:

Theorem 1:
For distributed k-means and k-median clustering on a graph, there exists an algorithm that with probability at least $1 - \delta$, the union of its output on all the nodes is an ϵ-coreset for P = $\bigcup_{i=1}^{n} P_i$. The communication cost is O(mn) and the following are the sizes for the coresets obtained for the two variants:
k- means: $O(\frac{1}{\epsilon^4}(kd + \log\frac{1}{\delta}) + nk\log\frac{nk}{\delta})$
k-medians: $O(\frac{1}{\epsilon^2}(kd + \log\frac{1}{\delta}) + nk)$.

In order to construct an algorithm that satisfies Theorem 1, the following parameters are defined:

1. **Sampling Probability:** This is the probability with which data points are chosen to be candidates for coresets and is defined as
$$Pr[q=p] = \frac{m_z}{\sum_{z \in P} m_z}$$
where m_p is derived based on the function applied on points.
2. **Weights on coreset points:** The coreset to be constructed consists of two types of points - points (S) sampled based on the probabilities described above and the centers of approximation at local nodes (B). The weights on these points are obtained as follows:
$$w_q = \frac{\sum_i \sum_{z \in Pi} m_z}{m_P\,|S|} \text{ for } q \in S \text{ and } w_b = |P_b| - \sum_{q \in P_b \cap S} w_q \text{ for } b \in B.$$
Here, P_b denotes the number of points associated with center b in the local approximation at any particular node.
3. **Function applied on the points**: A suitably defined function $f(p)$ is selected. The function nature varies for k-means and k-medians.

Consider K-medians :

Lemma 1:

Fix a set F of functions $f\colon P \rightarrow R_{\geq 0}$. Let S be an i.i.d. sample drawn from P according to $\{m_p : p \in P\}$. For $p \in P$, define $w_p = \frac{\sum_{z \in P_i} m_z}{m_p\,|S|}$. If for a sufficiently large c, we have $|S| \geq \frac{c}{\epsilon^2}$ (dim(F, P) + log $\frac{1}{\delta}$), then with a probability at least $1-\delta$, $\forall f \in F$, we have

$$\left| \sum_{p \in P} f(p) - \sum_{q \in S} w_q f(q) \right| \leq \epsilon\Big(\sum_{p \in P} m_p\Big)\Big(max_{p \in P} \frac{f(p)}{m_p}\Big)$$

Insights:

To get a small bound, set $m_p = max_{f \in F} f(p)$. By defining $f_x(p) = d(p,x) - d(b_p,x) + d(b_p,p)$ for every x, a provable upper bound can be obtained as $m_p = 2d(b_p,p)$. This is subsequently used to prove Lemma 2.

Lemma 2: For k-median, the output of the required algorithm is an $\epsilon - coreset$ with probability at least , if $t \geq \frac{c}{\epsilon^2}(dim(F,P) + log\,(\frac{1}{\delta})$ for a sufficiently large constant c.

Insights:

We have

$$D = \left| \sum_{p \in P} f_x(p) - \sum_{q \in S} w_q f_x(p) \right| \leq \epsilon \sum_{p \in P} m_p = 2\epsilon \sum_{p \in P} d(b_p,p) = 2\epsilon \sum_{p \in P} d(P_i, B_i) \leq O(\epsilon) \sum_{p \in P} d(p,x)$$

Upon substituting the definition of $f_x(p)$ as described in Lemma 1, in the expression for D and on subsequent simplification, it can be shown that

$D = \left| \sum_{p \in P} f_x(p) - \sum_{q \in S} w_q f_x(p) \right| = \left| \sum_{p \in P} d(p,x) - \sum_{q \in S} w_q d(q,x) \right| \leq O(\epsilon) \sum_{p \in P} d(p,x)$, which is a guarantee that the coreset obtained is an $\epsilon - coreset$ for the data.

Consider k-means:

Note that the above mentioned lemma 2 holds true for k-means as well when t=O($\frac{1}{\epsilon^4}$ (kd + log $\frac{1}{\delta}$) + nk log $\frac{nk}{\delta}$) i.e. an ϵ-coreset with probability at least (1- δ) can be constructed using Distributed_Coreset_Construction explained in the pseudo-code.

Key idea for the proof:

As an approximation to the original data points p, b_p centers from the local approximation solutions are used to prove that the error between the total cost and the weighted sample cost is approximately the error between the cost of p and its sampled cost.

Next, using Lemma 1, it is known that this difference between the cost is small, which proves the result.

Proof for k-means:
Now, the proof replicates the ideas presented above for the proof for k-median case, but the major difference is that triangle inequality cannot be applied directly to the k-means cost. For k-means, the error in the costs, $|\text{cost}(p,x) - \text{cost}(b_p, x)| = | d(p,x)^2 - d(b_p,x)^2 |$ does not have an upper bound as with k-median proof.

To tackle the issue, we divide the points in two categories, namely good points when the costs approximately satisfy the triangle inequality and the bad points:

- Define the good points with centers x as $G(x) = \{p \in P : |cost(p,x) - cost(b_p, x)| \leq \Delta_p\}$ where $\Delta_p = \frac{cost(p, b_p)}{\epsilon}$. These points can be bound as before.
- For bad points, it is proven that the difference in cost may be larger than $\frac{cost(p, b_p)}{\epsilon}$ but will be $O(\epsilon min\{cost(p,x), cost(b_p, x)\})$.

Next, $f_X(p)$ are defined only over good points as follows:
$f_X(p) = cost(p,x) - cost(b_p, x) + \Delta_p$ if p $\in G(x)$, 0 otherwise

Then $\sum_{p \in P} cost(p,x) \quad \sum_{q \in S \bigcup B} w_q cost(q,x)$ can be decomposed into the following three terms:

$$\sum_{p \in P} f_X(p) - \sum_{q \in S} w_q f_X(q) \qquad \text{--- (3)}$$

$$+ \sum_{p \in P \setminus G(x)} [\, cost(p,x) - cost(b_p, x) + \Delta_p \,] \qquad \text{--- (4)}$$

$$- \sum_{q \in S \setminus G(x)} w_q [\, cost(q,x) - cost(b_q, x) + \Delta_q \,] \qquad \text{--- (5)}$$

Now, considering these equations separately:

- Equation (3) can be bounded by $O(\epsilon) cost(P, x)$ by Lemma 1.
- Equation (4) can be bounded by $O(\epsilon) \sum_{p \in P \setminus G(x)} cost(p,x) \leq O(\epsilon) cost(P, x)$.
- Equation (5) is bounded by $O(\epsilon) \sum_{p \in P} cost(b_p, x)$.

Since each (3), (4), (5) is bounded by $O(\epsilon) cost(P, x)$, the sum is the same magnitude. Combining the above bounds, we have $| \sum_{p \in P} cost(p,x) - \sum_{q \in S \bigcup B} w_q cost(q,x) | \leq O(\epsilon) cost(P, x)$.

Hence, by choosing an appropriate ϵ, and bounding dim(F, P) = O(kd), the result is established.
Hence Proved.

The following theorem summarizes the distributed clustering algorithm results for a general graph.

Theorem 2:
Given an $\alpha -$ approximation for the weighted k-means and k-median subroutine, there exists an algorithm with probability at least $1 - \delta$ outputs a $(1+\epsilon)\alpha$ - approximation solution for distributed k-means and k-median algorithm on a rooted tree of height h.
The communication cost for two variants are as follows:
k- means: $O(m \frac{1}{\epsilon^4} (kd + \log \frac{1}{\delta}) + nk \log \frac{nk}{\delta})$
k-median: $O(m \frac{1}{\epsilon^2} (kd + \log \frac{1}{\delta}) + nk)$.

Next, the paper considered a specific case for rooted trees and examined the results for the same, which are summarized below.

Special Case: (*Rooted Trees*)
The algorithm can also be applied to the rooted trees and using experiments, it is proved that it compares well to other approaches involving coresets. The following theorem summarizes the results for the case of rooted trees.

Theorem 3:
Given an $\alpha -$ approximation for the weighted k-means and k-median subroutine, there exists an algorithm with probability at least $1 - \delta$ outputs a $(1+\epsilon)\alpha$ - approximation solution for distributed k-means and k-median algorithm on a rooted tree of height h.
The communication cost for two variants are as follows:
k- means: $O(h \frac{1}{\varepsilon^4} (kd + \log \frac{1}{\delta}) + nk \log \frac{nk}{\delta})$
k-median: $O(h \frac{1}{\varepsilon} (kd + \log \frac{1}{\delta}) + nk)$.

Pseudo-code:

```
def Message_Passing( $I_i$, $N_i$ ):
        # $I_i$ is the message, $N_i$ are the neighbors
        Let $R_i$ denote the information received
        Initialize $R_i = \{ I_i \}$, and send $I_i$ to all neighbors $N_i$
        While $R_i \neq \{ I_j, 1 \leq j \leq n \}$:
                If received message $I_j \notin R_i$:
                        $R_i = R_i \cup \{ I_j \}$ and send $I_j$ to all neighbors

def Distributed_Coreset_Construction( Local Datasets { Pi, $1 \leq i \leq n$ }, t, { $N_i$ } ):
        # t is the number of points to be sampled, $N_i$ are the neighbors of node $v_i$
        Round 1:
        On each node $v_i \in V$:
                Find set of centers $B_i$ for $P_i$
                Compute a constant approximation $B_i$ for $P_i$
                Communicate cost to all other nodes using Message_Passing( cost( $B_i$, $P_i$ ), $N_i$ )
```

```
    Round 2:
    On each node $v_i \in V$:
        Set $t_i = \frac{t\,cost(Pi, Bi)}{\sum_{j=1}^{n} cost(Pj, Bj)}$ and $m_p = 2cost(p, Bi)\ \forall p \in P_i$
        # where $t_i$ is the number of points to be sampled from node $v_i$
        # and $m_p$ is the weight of point p in $P_i$
        Pick a non-uniform random sample $S_i$ of $t_i$ points from $P_i$, where for every $q \in P_i$,
        we have q = p with probability $\frac{m_p}{\sum_{z \in Pi} m_z}$
        Let $w_q = \frac{\sum_i \sum_{z \in Pi} m_z}{t m_q}$ for each $q \in S_i$
        $\forall b \in B_i$, let $P_b = \{ p \in P_i : d(p, b) = d(p, B_i) \}$, $w_b = |P_b| - \sum_{q \in P_b \cap S} w_q$
    Return: Distributed coreset: points $S_i \cup B_i$ with weights $\{ w_q : q \in S_i \cup B_i \}$, $1 \le i \le n$

def Distributed_Clustering_on_Graph( { P_i }, { N_i }, A_α ):
    # $\{ P_i \}$ are the local datasets, $1 \le i \le n$; $\{ N_i \}$ are the neighbors of $v_i$, $1 \le i \le n$;
    # $A_\alpha$ is an $\alpha$-approximation algorithm for weighted clustering instances
    Round 1:
    On each node $v_i$
        Construct its local portion $D_i$ of an $\epsilon/2$-coreset by
        Distributed_Coreset_Construction( Pi, t, $N_i$ )
    Round 2:
    On each node $v_i$
        Call Message_Passing( $D_i$, $N_i$ )
        $\mathbf{X} = A_\alpha( \bigcup_j D_j )$
    Return: X
```

Experiments and Results:

Author has theoretically proved that his proposed algorithm has better bounds on communication cost. Exhaustive experiments were performed to compare this algorithm to other distributed corset algorithms. In the experiments the k-means cost of the solutions produced by proposed algorithm (with varying communication cost) was compared to other algorithms when they use same amount of communication.

Experimental Methodology:

A communication graph was first created connecting local sites and data was partitioned later into local datasets. Algorithm was evaluated on several network topologies (random/preferential/grid) and partition method (uniform, similarity-based, and weighted). When the network is a grid graph, author considered the similarity-based and weighted partitions. When the network is a preferential graph, author considered the degree-based partition. To measure the quality of coresets generated, Lloyd's algorithm was run on coresets and global data, and ratio between the cost of two solutions over global data was computed. The proposed

algorithm was compared with the naïve method of combining a coreset of each local dataset and the algorithm of Zhang et al.[1].
All the datasets used are available at University of California Irvine Machine Learning Repository [3].
Following table represents the topologies and partition method used on datasets :

Sr. no	Name	Number of Observations	Dimensions	Number of clusters	Topology
1	Spam dataset	4601	58	10	Random/Preferential/grid (10 sites, 3 x 3 grid graph)
2	Pendigits dataset	10992	16	10	Random/Preferential/grid (10 sites, 3 x 3 grid graph)
3	Corel image dataset	68040	32	10	Random/Preferential/grid (25 sites, 5 x 5 grid graph)
4	Letter dataset	2000	16	10	Random/Preferential/grid (10 sites, 3 x 3 grid graph)
5	YearPredictionMSd dataset	515345	90	50	Random/Preferential/grid (100 sites, 10 x 10 grid graph)

Table 1 : Datasets and their attributes

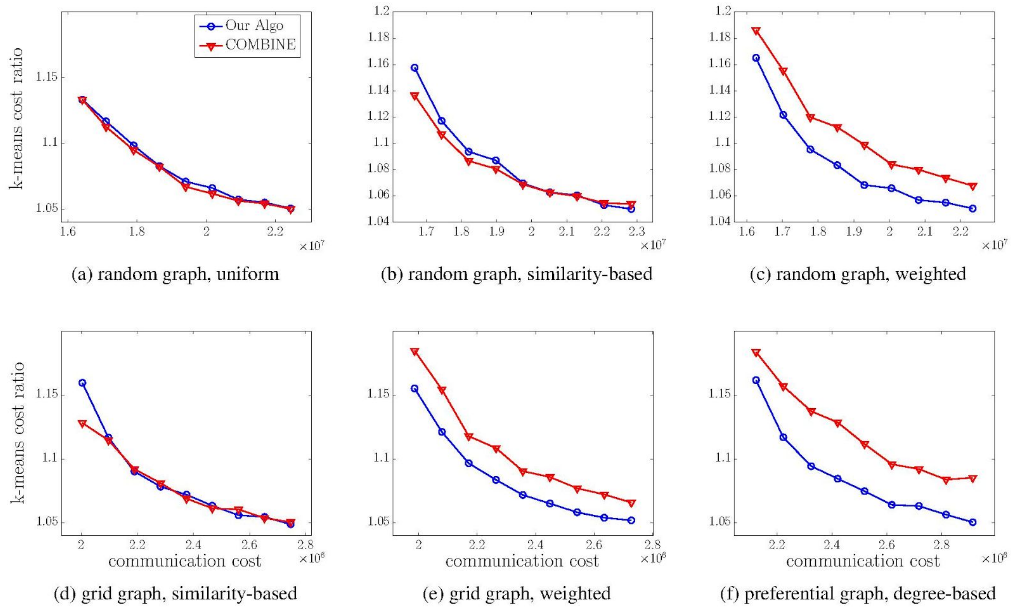

Figure 1: k-means cost (normalized by baseline) v.s. communication cost over graphs. The titles indicate the network topology and partition method.

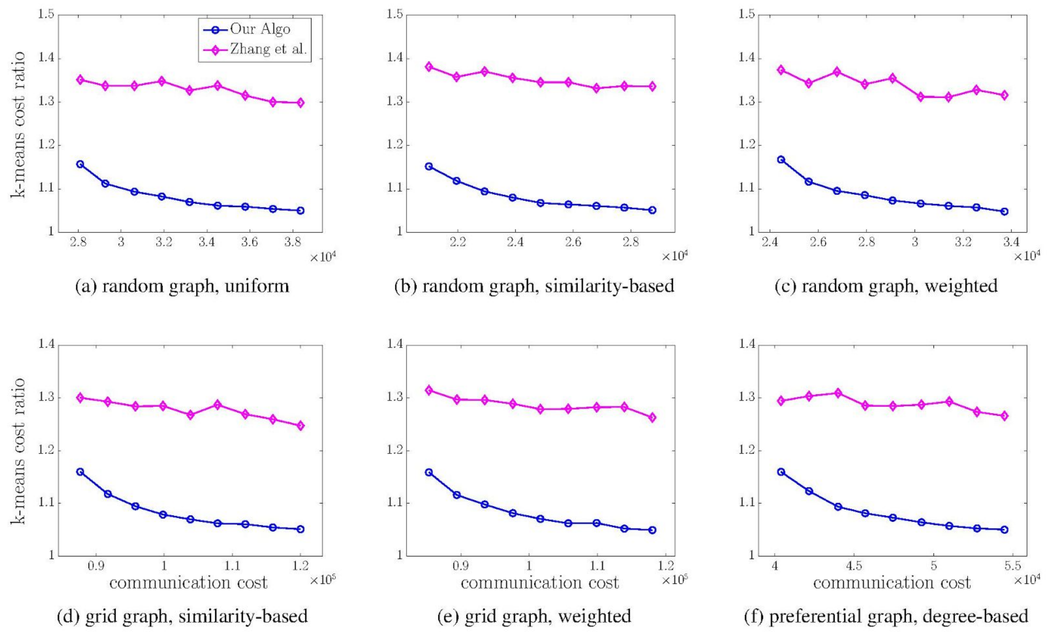

Figure 2 : k-means cost (normalized by baseline) v.s. communication cost over the spanning trees of the graphs. The titles indicate the network topology and partition method.

Results :

Results of the largest dataset, YearPredictionMSD are discussed in the paper. Figure 1 shows the result over different network topologies and partition methods.We observe that the algorithms perform well with much smaller coreset sizes than predicted by the theoretical bounds. Figure 2 shows the results over the spanning trees of the graphs. The proposed algorithm performs much better than Zhang et al.[26], achieving about 20% improvement in cost.

Dataset and Application:

Based on this research paper code was written to generate coresets in distributed setting. The ratio of cost for k-means on coresets and global data was studied for different communication cost. Three datasets from University of California- Irvine machine learning repository were used

1) Corel Image Features : This dataset contains image features extracted from a Corel image collection. Four sets of features are available based on the color histogram, color histogram layout, color moments, and cooccurence texture. From each image four sets of features were extracted: Color Histogram, Color Histogram Layout, Color Moments, Co-occurrence Texture. We used data on color histogram for clustering.
   HSV color space is divided into 32 subspaces (32 colors : 8 ranges of H and 4 ranges of S). The value in each dimension in a Color Histogram of an image is the density of each color in the entire image. Histogram intersection (overlap area between Color Histograms of two images) can be used to measure the similarity between two images. There are total 68040 observations.
2) Spambase Data Set : There are total 4601 observations with 58 dimensions. There were 1813 marked as spam and 2788 marked as non-spam. These attributes are of type repetition of certain words or average length of uninterrupted sequences of capital letters.

3) Letter Recognition Dataset : The dataset was made to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet.The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file of 20,000 unique stimuli. Each stimulus was converted into 16 primitive numerical attributes (statistical moments and edge counts) which were then scaled to fit into a range of integer values from 0 through 15. There are total of 20000 observations each observation has 16 attributes.

The fourth dataset used was collected from San Francisco Municipal Transportation Agency under FOIL - Freedom of Information Law.

1) SFpark dataset - The dataset contains data from three hundred parking lots in San Francisco city fitted with Smart Meters. There were total 8 attributes like session start date, session end date, Payment type, Net Amount paid, Post ID, Date, Parking district and Street and block address. There were about 1.6 million observations.It represented parking usage in about 300 parking lots in the month of April 2013. This data was preprocessed to and made suitable for clustering application. In the processed data the rows represents Parking lots and columns represents time. There was one column representing a 15 min slot. There were total 300 rows (one for each parking lot) and 2880 columns (24 hours x 15 time slots x 30 days).

Application:

With growing volumes of data it is getting harder day by day to fit it into the memory of a single computer. Furthermore, in many of these applications the data is inherently distributed because it is collected at different sites. So, many researchers and companies are using distributed settings to handle these large volumes of data. Some of the new advancements in applications of distributed clustering are as follows.[4] 1) Environmental monitoring applications, 2) Military and Surveillance applications, 3) Healthcare application.

References :

[1] Q. Zhang, J. Liu, and W. Wang. Approximate clustering on distributed data streams. In Proceedings of the IEEE International Conference on Data Engineering, 2008.
[2] D. Feldman, A. Sugaya, and D. Rus. An effective coreset compression algorithm for large scale sensor networks. In Proceedings of the International Conference on Information Processing in Sensor Networks, 2012.
[3] K. Bache and M. Lichman. UCI machine learning repository, 2013.
[4] S.R.Boselin Prabhu, S.Sophia, S.Maheswaran, M.Navaneethakrishnan. Real-World Applications of Distributed Clustering Mechanism in Dense Wireless Sensor Networks, 2013.

Submitted by:

Aishwarya Anandan (aananda2), Harshad Rai (hrrai2), Rachneet Kaur (rk4), Vipul Satone(vsatone2)