Clustering Dynamic and Distributed Dataset using Decentralized Algorithm

Mary Femy P.F[1] Linda Sara Mathew[2]
[1,2]Department of Computer Science & Engineering
[1,2]Mar Athanasius College of Engineering, Kothamangalam, Kerala, India

Abstract— In many popular applications large amounts of data are distributed among multiple sources. Analysis of this data and identifying clusters is challenging due to storage, processing, and transmission costs. A decentralized clustering algorithm called DCluster, which is capable of clustering distributed and dynamic data sets. Nodes continuously cooperate through decentralized gossip-based communication to maintain summarized views of the data set. The summarized view is a basis for executing the clustering algorithms to produce approximations of the final clustering results. DCluster can cluster a data set which is dispersed among a large number of nodes in a distributed environment. In DCluster the complete data set is clustered in a fully decentralized fashion, such that each node obtains an accurate clustering model, without collecting the whole data set.

Key words: DCluster, Distributed dataset, Dynamic

I. INTRODUCTION

Identify clusters, is an important factor in the analysis of large datasets. Generally, for extracting data, eliminating duplicate data, and making usable these data, several techniques have been proposed as data mining methods. As a result, data mining has emerged as an important area of research. Distributed computing environments have separated and diffuse data sources. Due to the large volume of computing and communications and network bandwidth limitations, privacy reasons, or because of the huge amount of distributed data, it's essential that the processing of data be performed using a distributed approach, without aggregate data to a centralized location. Unsupervised clustering is a popular learning task with many application fields such as data compression, computer vision, and data mining. Available contents in those fields are growing exponentially and with no doubt faster than the computing performances of individual machines. Besides, data tends to originate more and more often from decentralized sources.

Most classic clustering algorithms are designed for the centralized setting, but in recent years data has become distributed over different locations, such as distributed databases, videos and images over networks and sensor networks. In many of these applications the data is inherently distributed because, as in sensor networks, it is collected at different sites. As a consequence it has become crucial to develop clustering algorithms which are effective in the distributed setting.

Distributed data clustering, aims to extract potentially useful information from large datasets by grouping similar data, and separating dissimilar data according to some criteria of dissimilarity between data items. In a distributed environment, it needs to be done when the data cannot be concentrated on a single site, for example, for reasons of security concerns or due to bandwidth limitations or due to high volumes of distributed data.

Clustering is a well-known and widely used exploratory data analysis technique. Most of the clustering algorithms that are available in the literature deal with data available at a single location. However, there exist many applications where data sources are distributed over a network and collecting the data at a central location before clustering is not a viable option. Decentralized Clustering algorithm (DCluster) can cluster a data set which is dispersed among a large number of nodes in a distributed environment. It can handle partition-based clustering, while being fully decentralized, asynchronous, and also adaptable to churn.

II. PROPOSED MODEL

The proposed Decentralized Clustering (DCluster) model is shown in Figure 1. It consider a set P = (p1; p2; . . . ; pn) of n networked nodes. Each node p stores and shares a set of data items Dintp , denoted as its internal data, which may change over time. D = UpϵP Dintp is the set of all data items available in the network. Each data item d is presented using an attribute (metadata) vector denoted as dattr. While discovering clusters, p may also store attribute vectors of data items from other nodes. These items are referred to as the external data of p, and denoted as Dextp . The union of internal and external data items of p is referred to as Dp. During algorithm execution, each node p gradually builds a summarized view of D, by maintaining representatives, denoted as Rp = (rp1; rp2; . . . ; rpkp). Each representative r ϵ Rp is an artificial data item, summarizing a subset Dr of D. The attribute vector of r, rattr, is ideally the average of attribute vectors of data items in Dr. The intersection of these subsets need not be empty.

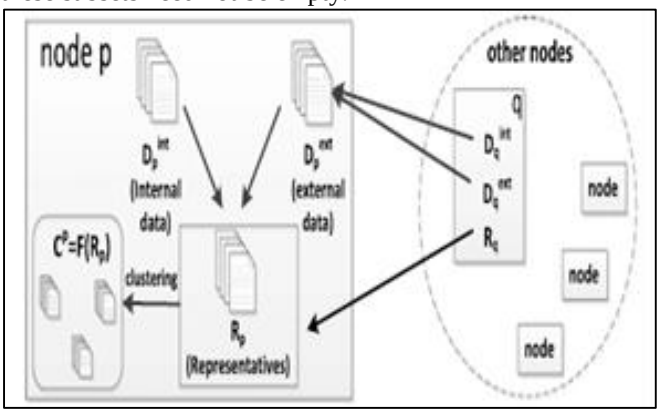

Fig. 1: Decentralized clustering model

The entire data set can be summarized in each node p, by means of representatives. Each node p is responsible for deriving accurate representatives for part of the data set located near $D^{int}p$. For other parts, it solely collects representatives. Accordingly, it gradually builds a global view of D. Each node continuously performs two tasks in

parallel: i) Representative derivation[1], DERIVE and ii) representative collection[1], COLLECT. The two tasks can execute repeatedly and continuously in parallel.

To derive representatives for part of the data set located near D^{int}_p , p should have an accurate and up-to-date view of the data located around each data d ϵ D^{int}_p . In each round of the DERIVE task, each node p selects another node q for a three-way information exchange. It should first send D^{int}_p to node q. Node p then receives from q, data items located in radius ρ of each d ϵ D^{int}_p , based on a distance function. ρ is a user-defined threshold, which can be adjusted as p continues to discover data. In the same manner, it will also send to q the data in Dp that lie within the ρ radius of data in D^{int}_q. Knowing some data located within radius ρ of some internal data item d, node p can summarize all this data into one representative.

To fulfill the COLLECT task, each node p selects a random node every T time units, to exchange their set of representatives with each other. Both nodes store the full set of representatives. Initially, each node has only a set of internal data items, D^{int}_p. Thus, the set of representatives at each node is initialized with all of its data items, i.e., Rp = D^{int}_p.

The final clustering algorithm is executed on the set of representatives in a node. Node p can execute the clustering algorithm on Rp, any time it desires, to achieve the final clustering result. In a static setting, continuous execution of DERIVE and COLLECT will improve the quality of representatives causing the clustering accuracy to converge. K-means[8] considers data items to be placed in an m-dimensional metric space, with an associated distance measure. It partitions the data set into k clusters, C1; C2; . . . ; Ck. Each cluster Cj has a centroid μj, which is defined as the average of all data assigned to that cluster. The K-means algorithm is executed on a set of representatives, each extracted from data within ρ distance of a data item, and its ultimate goal at node p is to compute the mean of data in each cluster. Let D_{Ci} denote the data items of a typical cluster Ci, and R_{Ci} denote representatives computed from data in D_{Ci}. If D_{Ci} is uniform, the expected value of the representatives will be equal to μi.

III. RESULTS

Interface provides the facility to select the data set and to give the number of nodes in the system. Figure 2 shows the main GUI for general distributed clustering. It also provides the facility to set the output file location and to add extra data to the system.

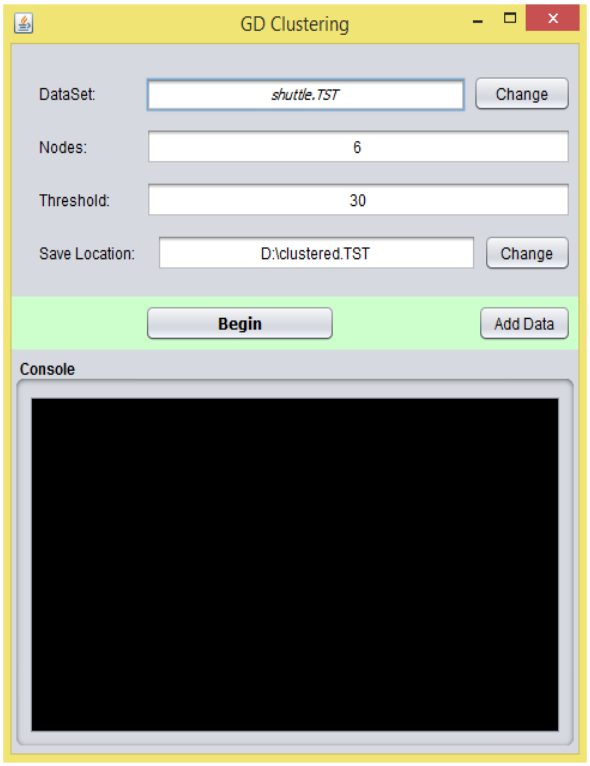

Fig. 2: DCluster GUI

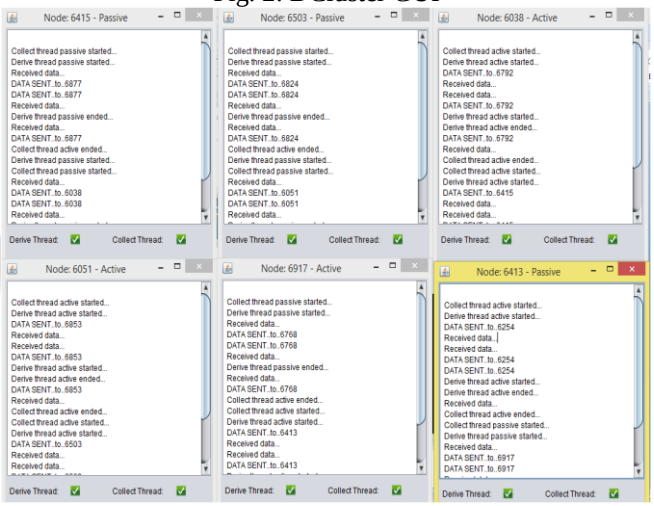

Fig. 3: Node Interface

Fig. 3 shows the interface for nodes. This shows status of various tasks in each node, such as derive thread active started/ended, derive thread passive started/ended, data sent details, data receiving details, collect thread active/passive started/ended. Fig.4 shows the output file. It consists of centroids of each cluster.

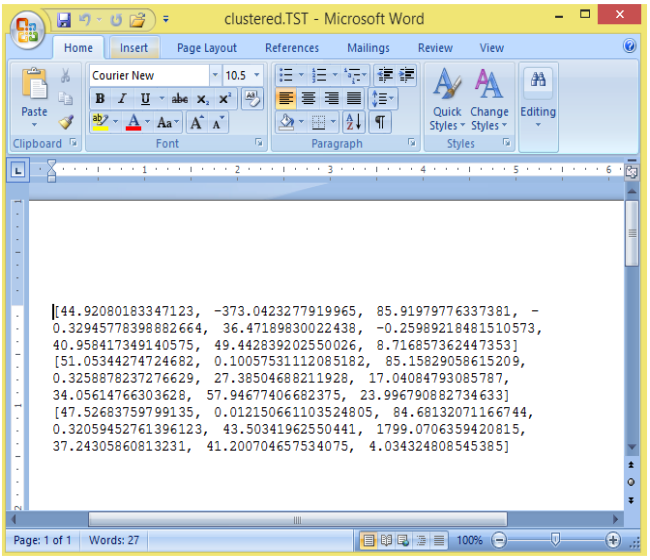

Fig. 4: Output File

IV. Performance Analysis

In order to assess the efficiency of algorithm in detecting clusters, compare its outcome to that of centralized K-means. Executing K-means centrally on a given data set results in a set of clusters *C1; C2; . . . ; Ck*, which will be referred to as real clusters. Likewise, at any time while executing the algorithm, each node *p* can derive a set of clusters *Cp1; Cp2*; *. . . ; Cpk* , which will call computed clusters of node *p*. Mapping function map(c), maps a computed cluster c to some equivalent real cluster. Each data item $d \in D$, belongs to a specific global cluster *C(d)*, and a specific computed cluster in each node *p*, denoted as *Cp(d)*.

The performance of DCluster in terms of accuracy for different number of internal data (Nint) is given in Table I. It is performed in four nodes with number of cluster four.

N_{int}	Σ eq(C(d), map(C^p(d)))	AC (%)
50	74	37
100	214	53.55
150	431	71.83
200	642	80.25

Table 1: Accuracy Analysis

The Fig. 5 shows the performance of DCluster for different number of internal data. From the results it is clear that when nodes have few data, detecting accurate clusters is harder, due to sparseness of clusters.

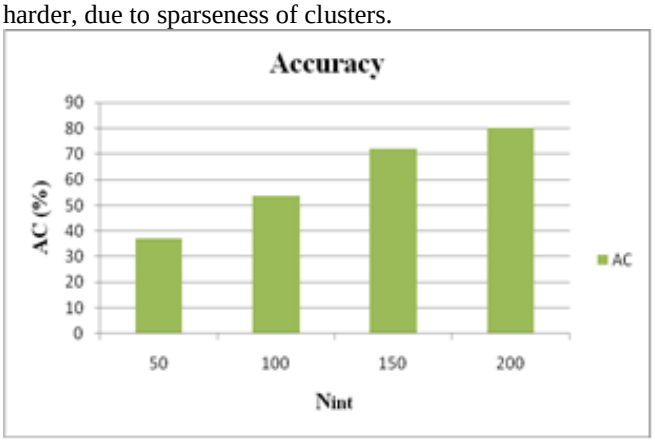

Fig. 5: Accuracy

V. Conclusion

Clustering partitions data into groups of similar objects, with high intra-cluster similarity and low inter-cluster similarity. With the progress of large-scale distributed systems, huge amounts of data are increasingly originating from dispersed sources. Analyzing this data, using centralized processing, is often infeasible due to communication, storage and computation overheads. DCluster is a general distributed clustering method, which is capable of clustering dynamic and distributed data sets in a decentralized manner. Nodes continuously cooperate through decentralized gossip-based communication to maintain summarized views of the data set. Dynamic nature of data demands a continuously running algorithm which can update the clustering model efficiently, and at a reasonable pace. This algorithm enabled nodes to gradually build a summarized view on the global data set, and execute clustering algorithms to build the clustering models. The final clustering algorithm is executed on the set of representatives in a node. DCluster can be customized for other clustering types such as hierarchical or grid-based clustering.

References

[1] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A Density based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," In Proceedings of Knowledge Discovery in Database (KDD), pp. 226–231, 1996.

[2] A.K. Jain, M.N. Murty, P. J. Flynn, "Data Clustering: A Review," ACM Computing Surveys, Vol. 31(3), pp. 265-323, 1999.

[3] S. Datta, C. Giannella, H. Kargupta, "K–Means Clustering over a Large, Dynamic Network," Proc. SIAM Int'l Conf. Data Mining, pp. 153-164, 2006.

[4] J. Han, and M. Kamber, "Data Mining: Concepts and Techniques," 2nd ed, Morgan Kaufmann Publishers, 2006.

[5] M. Li, G. Lee, W.C. Lee, and A. Sivasubramaniam, "PENS: An algorithm for Density-Based Clustering in Peer-to-Peer Systems," Proceedings of the 1st international conference on Scalable information systems, pp. 39, 2006.

[6] S. Datta, C. Giannella, and H. Kargupta, "Approximate distributed kmeans clustering over a peer-to-peer network", IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 10, pp. 1372–1388, 2009.

[7] S. Lodi, G. Moro, and C. Sartori, "Distributed data clustering in multi-dimensional peer-to-peer networks," in Proc. 21st Australasian Conf. Database Technol., vol. 104, pp. 171–178, 2010.

[8] Elgohary and M. A. Ismail, "Efficient data clustering over peer to peer networks," in Proc. 11th Int. Conf. Intell. Syst. Des. Appl., 2011, pp. 208–212.

[9] K. M. Hammouda and M. S. Kamel, "Models of distributed data clustering in peer-to-peer environments," Knowl. Inf. Syst., vol. 38, no. 2, pp. 303–329, 2014.

[10] Hoda Mashayekhi, Jafar Habibi, Tania Khalafbeigi, Spyros Voulgaris, and Maarten van Steen, "GDCluster: A General Decentralized Clustering Algorithm," Knowl. Data. Engg., vol.27, no.7, 2015.

