PROPOSED RESEARCH:

The maximum length of this file is **10 pages** (letter size, Verdana size 10 is suggested)**.** For an adequate evaluation of your proposal merits,this file must include the following aspects: Proposal description, Hypothesis,Goals, Methodology, Work Plan, Work in progress and Available Resources. Make sure to describe the relevance of the proposal topic in relation to the state of the art in the field. Keep in mind the Bases del Concurso FONDECYT de Postdoctorado 2017 and Application Instructions.

1 Introduction

As a consequence of the explosive growth of the WEB, the integration of search engines such as Google into personal computers and mobile devices, and the wide use of social networks, the task of clustering text data, i.e. Tweets or documents in a search engine, has each time an increasing importance because of the necessity of unraveling the underlying categories in large volumes of text data. Nowadays the generation of large amounts of documents surpasses the computational power of personal computers and even of high performance computers. As an example, it is estimated that the amount of WEB pages that popular search engines such as Yahoo! and Google index is higher than the tens of a billion. It is therefore of great interest to develop algorithmic techniques capable of automatically organize, classify and summarize document collections distributed in multiple machines of a network, and that also perform efficiently in modern hardware, particularly within parallel processing frameworks in multi-core architectures. In real problems such as *Collection Selection* for distributed document databases, in which for a given query the computer node containing the most suitable sub-collection must be selected to answer it Crestani and Markov (2013), the challenges related to the scalability and efficiency of *Knowledge Discovery* methods have become very important. Traditional algorithms often assume that the whole dataset is loaded into main memory (RAM) and thus every document can be accessed at any time with no access latency. In scenarios where the size of the collection is much bigger than the amount of available RAM either because of the number of documents or the length of each document, this ideal loading process is unfeasible due to real constraints in the storage or computational capabilities.

Often, text data is structured in digital collections of documents whose length (number of characters) is variable, e.g. WEB pages or the content generated by users in social networks such as Twitter or Facebook. In order to enable the processing of these collections, in a first stage of preprocessing a set of words occurring in the documents are extracted and sorted in lexicographical order; this word set is referred to as the Vocabulary. Then, the content of every document in the collection is represented as a vector in which each dimension denotes a specific word of the Vocabulary and its value in a document vector is given by a function of the number of occurrences of the word within the document and the number of documents in which it appears. As a natural consequence of the lexical richness of every language, the size of the Vocabulary, and in turn the dimensionality of the vector space onto which a document is represented, is far bigger than the data size that traditional clustering algorithms manage. Because of this, the task of automatic document clustering has high computational and storage (RAM and secondary memory) costs. Along with this, when the number of documents is large (tens or hundreds of thousand and even millions of items) then traditional techniques for processing and clustering documents, and current computational capabilities of a single machine and also high performance machines are insufficient. Even in the most favorable scenario the storage and computational power tackle the challenge but the response time are excessive.

There exist three approaches successfully applied to the construction of clustering algorithms capable of processing large volumes of data. The first one introduces constraints on the number of passes allowed on a document (related to the amount of time it is loaded into main memory). The second one exploits current multi-core architectures to perform parallel processing of the data, which does not solve the massive volume problem. The last one combines the computational power of a single machine together with the scalable storage capability of a distributed system by partitioning the dataset into several independent machines connected through a network.

The last above mentioned approach seems promising since it allows to exploit the local capabilities of single computers with multi-core architectures without sacrificing scalability to large data volumes because of its distributed design. Within this path there are two contexts regarding the data generation scenario. On the one hand, in some problems where the dataset is large but collected in a centralized fashion, the strategy employed consists in partitioning the collection into several machines or nodes of a network. This scheme leads to the transmission of a lot of data during the execution of the algorithm Nagwani (2015). On the other hand, there are some problems where the data is generated in a distributed fashion and where besides it is not feasible to centralize the data because of high transmission costs or privacy issues Jagannathan et al. (2005); Liu et al. (2012), e.g. search engines work with document collections originated and stored in different geographical locations.

2 Problem Definition and State of the Art

Let $X = \{X_1, X_2, \ldots, X_p\}$ be a collection of documents where $\bigcup_{i=1}^{p} X_i = X$ and $\forall i \neq j \in [1, \ldots, p], X_i \cap X_j = \emptyset$. This collection is partitioned into p different machines of a network, and also each subset X_i consists of n_i vectors in $\mathbb{R}^d$. Additionally, in a distributed environment, data sites may be homogeneous or heterogeneous, depending if different sites contain data for exactly the same set of features or for different set of features respectively. The distributed data clustering task consists in

obtaining k data partitions $C_1, C_2 \ldots, C_k$ where $\bigcup_{i=1}^{k} C_i = X$ and also $\forall i \neq j \in [1, \ldots, k], C_i \cap C_j = \emptyset$, such that each one of them represents a topical category of the overall collection and in turn all the categories are covered by the partitions or clusters C_i. Expressed in a different manner, this means that every cluster contains more similar documents according to their content in comparison to the other ones contained in different clusters. The similarity between document vectors is usually measured by means of a function $S : \mathbb{R}^d \times \mathbb{R}^d \rightarrow [0, 1]$, e.g. the cosine similarity function:

$$S(x_a, y_b) = \frac{< x_a, x_b >}{||x_a||x_b|}$$

, where $< u, v >$ with $u, v \in \mathbb{R}^d$ denotes the inner product between two vectors and $|u|$ corresponds to the Euclidean norm of a vector. In a nutshell, this task consists in finding a structure of compact and homogeneous groups with respect to a similarity value between their members that accounts for their content.

An overall operation mode of the distributed data clustering techniques consists of four stages: Initially, the dataset is partitioned into several nodes. Next, each node generates a clustering model over its corresponding local subset. These local models are then transmitted to a central node (e.g. A model can consists of representative points identified from a local clustering), which integrates the local models in a global solution. Eventually, this global model can be re-transmitted to the other nodes in order to refine their local models and then, by performing the same previous steps, a new depurated global model is built. This scheme is depicted in Figure 1.

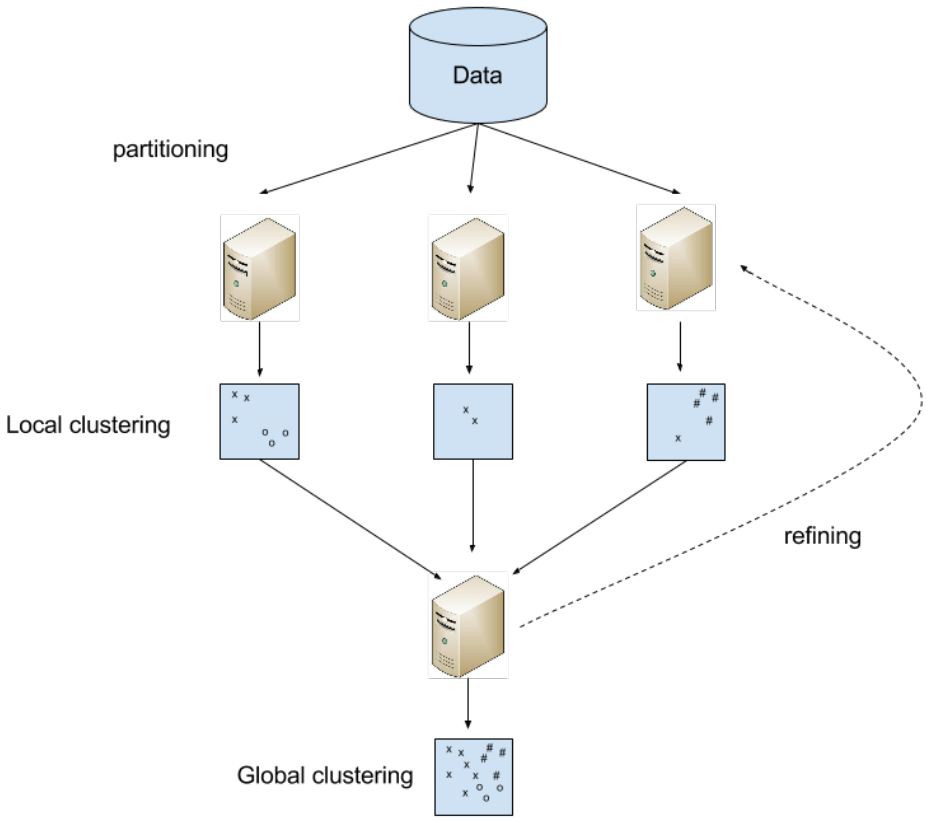

Figure 1: Overall scheme of operation of the distributed data clustering techniques.

2.1 Parallel and distributed approaches for massive and high dimensional data clustering

As far as we know from the literature, most of the existing efforts for the construction of clustering techniques capable of operating in scenarios where the data is distributed have been focused on low dimensional data (less than 100 attributes) in contrast with document datasets in which a document vector for a small collection may have about 10^4 attributes. Nevertheless, the main advances in distributed data clustering are detailed below, specially highlighting those contributions focused on methods capable of dealing with high dimensional data.

1. Main contributions on parallel algorithms Xu et al. (2002) and Dhillon and Modha (1999) propose parallel extensions for the algorithms DBSCAN and K-Means. Another scalable approach based on secondary memory consists in designing algorithms capable of working within the Mapreduce framework[1]. In this context it is possible to highlight the contributions made by Das et al. (2007) in which they propose an implementation of the EM algorithm and also by Ene et al. (2011) in which they tackle the K-Median problem by using Mapreduce. Another parallel approach for K-Means is presented by Bahmani et al. (2012) and it is called K-Means++. Additionally, the EM-Tree proposed by De Vries et al. (2015) is very interesting since it allows to process very large datasets, specifically the authors show that it can handle hundred of millions of WEB pages.

2. Approaches capable of dealing with high dimensional data Kargupta et al. (2001) propose a method to obtain the Principal Components (PCA) over heterogeneous and distributed data. Based on this contribution on dimensionality reduction they also propose a clustering method that works over high dimensional data. Once the global principal components are obtained by using the distributed method and transmitted to each node, local data are projected onto the components and then a traditional clustering technique is applied. Finally, a central node integrates the local clusters in order to obtain a global clustering model. Several years later, Liang et al. (2013) present another algorithm for principal components extraction over distributed data. To this end, each node computes PCA over its local data and transmit a fraction of them to a central node. This node uses the received components to estimate the global principal components, which are later transmitted to each node. After this, in every node, local data are projected onto the global components and the projected data are used for computing a coreset by means of a distributed algorithm. The global coreset built from the local projected data will be finally used to obtain a global clustering model.

   Li et al. (2003) propose an algorithm called D-CoFD for high dimensional and distributed data that also is capable of dealing with homogeneous and heterogeneous environments.

3. Density based approaches Januzaj et al. (2003) propose a distributed data clustering technique in which local nodes build models and transmit a set of representatives of each cluster to a central node. In this node a centralized clustering method is applied over the representatives and then the resulting model is re-transmitted to the local nodes to update their models. Klusch et al. (2003) propose a clustering technique based on density estimates. Januzaj et al. (2004) present a scalable version of DBSCAN that is also capable of operating over distributed collections. First, the best local representatives a selected depending on the number of points that each one represents and then those chosen points are sent to a central node. This central node clusters the received local representatives into a single new model, which is re-transmitted to the other nodes so they can improve their local group structure.

4. Approaches based on parametric models Merugu and Ghosh (2003) propose a clustering method that combines local parametric models, each one built on a single node, into a general one. The main contribution of this work consists in a method capable of dealing with distributed data that also considers the privacy of the local data in each node, since it transmits a summary of each local data and not raw points. Kriegel et al. (2005) present a technique that fits a Gaussian mixture model in each node using the EM algorithm. Finally, all these Gaussian mixture models are integrated into a general parametric model.

5. Approaches based on representative points Forman and Zhang (2000) extend centroid based techniques to identify groups over distributed data. Specifically, they extend K-Means, K-Harmonic-Means and EM. Qi et al. (2008) propose an approximate K-Median clustering technique that works over stresaming data, i.e. data is continuously collected. Balcan et al. (2013) address the distributed clustering problem by using centroid based techniques that use a novel coreset construction method that works for distributed data. Naldi and Campello (2014) tackle the problem of the parameter selection of the K-Means clustering algorithm by using evolutionary algorithms. Additionally, they propose two strategies inspired in evolutionary algorithms for clustering distributed data using centroids.

6. Hierarchical clustering approaches A hierarchical algorithm that works on distributed and heterogeneous data is presented by Johnson and Kargupta (2000). This method assumes that data is vertically partitioned, thus all nodes contain the same set of points, but characterized by different features. At the beginning, a dendrogram is generated in each node, and then it is transmitted to a central node. Finally, this node combines all the dendrograms into a global model. Jin et al. (2015) propose a hierarchical clustering algorithm for distributed data that builds the clusters by incrementally processing the data points. The authors re-state the hierarchical clustering problem as a Minimum Spanning Tree construction problem over a graph. In order to integrate several local models they also propose a technique for combining multiple minimum spanning trees, assuming that these trees were obtained from disjoint subgraphs of the complete original graph. This mixture procedure iterates until a single tree is obtained, which in turn denotes the hierarchical clustering originally pursued.

[1] Hadoop MapReduce is a software solution that enables the construction of applications capable of processing large amounts of data (e.g. Terabytes) in a parallel fashion over big computer clusters.

3 Hypothesis

A distributed master/slave approach for the design of single-pass and parallel clustering algorithms will enable the construction of methods capable of attaining results comparable to a centralized scheme, in terms of standard clustering performance measures such as Rand-score, Mutual Information and V-Measure, for high dimensional, massive and distributed document databases.

4 Goals

The general aim of this work consists in developing new parallel clustering techniques capable of dealing with large document databases that also can be stored in several nodes of a computer network. Additionally, this work comprises the following specific objectives:

- To develop parallel clustering scheme for document collections that also performs a single-pass over each document, i.e. access to secondary memory during the algorithm execution is restricted.
- To extend the parallel scheme proposed to environments where the document collection is distributed into several machines.
- To validate the proposed models with benchmark and real text data extracted from specialized sites such as UCI and NIST.
- To disseminate the obtained results and methods in this research in scientific journals registered in the ISI catalog.

5 Methodology

In order to acomplish the goals of this project the following general and specific activities are distinguished:

1. Study and discussion of the state of the art literature. Specifically, a constant and complete revision of the literature related to:
   - Parallel and distributed Clustering methods.
   - Theoretical and experimental aspects involved in the single-pass and parallel clustering methods for distributed data.
   - Hashing strategies for the efficient estimation of neighborhood over centralized and distributed data collections.
   - Clustering methods for massive data collections.
   - Distributed and parallel Machine Learning algorithms.
2. Colaboration and dissemination activities:
   - Seminar with topics about Clustering and Non-Supervised methods and their applications over problems involving distributed and massive data collections. Researchers, practitioners and students (undergraduate and postgraduate) will be included.
   - Participation in national and international conferences related to Machine Learning techniques, Intelligent Systems for distributed data and Knowledge extraction from large databases (at least one per year).
   - Establishing contact for cooperation with related research groups either in the national as in the international sphere.
3. Design and construction of the proposed methods. In addition and considering the previously posed hypothesis and its validation, the following steps are established:
   - Modeling and analysis of results under environments presenting distributed and high dimensional data.
   - Construction of the proposed models, that is to build an implementation in the Python or Julia language, of a single-pass Clustering algorithm capable of operating under scenarios in which the vectors, each one representing a document, are distributed into several computers (workers).
   - An empirical study of techniques for summarizing massive text volumes (e.g. Coresets, medoids) with the aim to generate more succint snapshots of the current state of the stream. This is of particular interest in a distributed scenario in which a message passing based synchronization is performed between workers and the communication cost is restrictive.
   - Descripción de las propiedades teóricas de los modelos: Durante esta fase nos enfocaremos en la generación de conocimiento centralizado a partir de modelos locales de Clustering.

4. Implementación y Optimización de los Algoritmos Propuestos Los algoritmos propuestos serán implementados en un nuevo lenguaje llamado JULIA (http://www.julialang.org). Desarrollaremos una estrategia compuesta de módulos capaces de ser evaluados e intercambiados independientemente. Durante esta fase de desarrollo, identificamos los siguientes pasos (los cuales pueden ser iterados): a) Construcción de una plataforma general para construir la arquitectura de los prototipos, b) Implementación de diversos escenarios distribuidos, c) Implementación de algoritmos del estado del arte con fines comparativos, d) Implementación de las técnicas de validación, e) Elaboración de soporte para clusters y tecnologías de Cloud Computing tales como MPI, OpenStack y Amazon EC2, con el fin de simular el problema en un ambiente real.

5. Diseño de los experimentos y validación de los algoritmos propuestos. Para validar los modelos propuestos, tenemos planificado usar tanto datos "benchmark" conocidos en el área, como datos provenientes de problemas reales de interés regional. Los datos de "benchmark" serán recolectados de sitios web de acceso publico del área de Máquinas de Aprendizaje. En la mayoría de los casos, los conjuntos de datos tienen asociada a cada documento una etiqueta que indica su verdadera clase. A las medidas de evaluación que comparan las etiquetas asignadas por el método de clustering con las verdaderas se les denomina medidas externas. Las medidas externas que usaremos con el fin de validar la propuesta son: Rand Score, Mutual Information, Homogeneity, Completeness y V-Measure. Para aquellos conjuntos de datos que no cuenten con etiquetas verdaderas, se usará la medida interna Silhouette Rousseeuw (1987), la cual cuantifica que tan bien diferenciados están los grupos encontrados por el método bajo evaluación. La etapa de validación será realizada mediante la corrida de 20 experimentos con particionamientos aleatorios para cada uno de los conjuntos de datos utilizados. Los métodos propuestos serán comparados con modelos del estado del arte en al menos 10 conjuntos de datos, tanto reales como sintéticos obtenidos de dos fuentes:

   - UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/) el cual es el repositorio más extenso de datos usados en Machine Learning,
   - Datos provenientes de bases de datos documentales como Tipster y 20-Newsgroups.

6 Plan de Trabajo

Las etapas fueron descritas en el punto 5 y son las siguientes:

- *Etapa 1*: Estudio y discusión de la literatura relevante para la propuesta.
- *Etapa 2*: Diseño de los modelos y algoritmos propuestos:
  1. Modelamiento de ambientes con patrones distribuidos.
  2. Construcción de los modelos propuestos.
  3. Descripción teórica de las capacidades y propiedades de los modelos.
- *Etapa 3*: Implementación y optimización de los algoritmos propuestos.
- *Etapa 4*: Diseño de experimentos y estrategia de validación de los algoritmos propuestos.
  1. Generación de conjuntos de datos sintéticos.
  2. Recolección de datos reales y sintéticos.
  3. Procedimiento de validación.
  4. Análisis comparativo.
- *Etapa 5*: Diseminación de los resultados obtenidos en este proyecto.

Durante el año 2017 definiremos el modelo de clustering distribuido desde el punto de vista de la extracción de representantes desde cada fuente que permitan una posterior identificación de los grupos subyacentes a toda la colección (Etapa 1). Concentraremos esfuerzos en el estudio y construcción de técnicas basadas en Coresets y en distancias derivadas de vecinos más cercanos compartidos (Etapa 2 y 3). Por último, evaluaremos empíricamente y compararemos los algoritmos propuestos (Etapa 4).

Durante el año 2018 concentraremos nuestros esfuerzos en el estudio teórico de propiedades de los algoritmos que permitan justificar su desempeño e identificar escenarios menos favorables para su operación. Este análisis facilitará la comprensión de la contribución de los algoritmos propuestos para su posterior diseminación (Etapa 5).

7 Trabajo en Progreso

Una enfoque inicial de exploración para el diseño y construcción de algoritmos de clustering de colecciones documentales distribuidas consiste en una adaptación del algoritmo de Ertöz et al. (2003) para un contexto distribuído. En este esquema se ataca primero la alta dimensionalidad de los vectores mediante medidas de distancia basadas en la cantidad de vecinos que comparten dos puntos cualquiera. Por otra parte, la gran cantidad de datos y el ruído se ataca mediante la selección de representantes, denominados *Core-points*. Para esto se utilizan dos parámetros, **Eps** y **MinPts**. Luego, un punto será representante solamente si comparte más de **Eps** vecinos con más de **MinPts** puntos. Finalmente, se etiquetan con el mismo número de grupo aquellos *Core-points* que se compartan más de **Eps** puntos, y luego el resto de los puntos recibe la etiqueta de su *Core-point* más cercano en término de vecinos cercanos compartidos. En el caso de que el *Core-point* más cercano esté a distancia menor que **Eps**, se identifica como ruído.

El esquema anterior fue pensado en un escenario centralizado en donde es posible calcular las distancias entre todos los pares de puntos. Para un conjunto de datos con n puntos, cada uno representado por un vector en $\mathbb{R}^d$, el costo de calcular estas distancias es $\mathrm{O}(n^2 * d)$ y de almacenarlas es $\mathrm{O}(n^2)$. Al considerar grandes volúmenes de datos representados además por vectores altamente dimensionales, resulta claro que el desempeño del método difícilmente escale.

El trabajo que actualmente estamos realizando considera que la colección se encuentra particionada aleatoriamente en un conjunto de nodos o máquinas. En cada uno de estos nodos se usa el esquema de Ertöz et al. (2003) para identificar *Core-points* y etiquetar los datos locales. Tomando algunas ideas propuestas para la construcción de core-sets distribuídos por Balcan et al. (2013), se realiza en cada máquina una selección de *Core-points* aleatoria con pesos inversamente proporcionales a la cantidad de puntos con la etiqueta del *Core-point* y luego se transmiten los puntos seleccionados aun nodo central. Es importante mencionar que, a diferencia del trabajo en curso que se describe en esta sección, las contribuciones en esta línea no consideran en general datos de alta dimensionalidad (vectores con miles de atributos) ni tampoco grupos que pueden tener formas no esféricas. De esta manera, serán seleccionados con mayor probabilidad aquellos *Core-points* ubicados en grupos más pequeños, buscando así representar a todos los grupos independientemente de que tan pequeños sean. La cantidad de puntos seleccionados es un parámetro expresado en términos de porcentaje y es precisamente este mecanismo el que permite atacar los problemas generados por los grandes volúmenes de datos. Así en el nodo central se realizará nuevamente un clustering usando la medida de distancia basada en vecinos más cercanos compartidos, pero únicamente sobre una fracción de todos los *Core-points*. El agrupamiento final contendrá los grupos de la colección completa, lo que corresponderá al resumen esperado de la colección.

Actualmente, hemos realizado experimentos sobre datos sintéticos con formas esfericas y no esfericas, obteniendo resultados interesantes. Para poder realizar afirmaciones concluyentes, aplicaremos el método sobre colecciones documentales reales.

8 Recursos disponibles

Los recursos disponibles para este proyecto en la Escuela de Ingeniería Informática de la Pontificia Universidad Católica de Valparaíso son:

- Biblioteca de la Universidad y biblioteca personal del investigador patrocinante.
- Salón de seminarios.
- Oficina, Internet, Teléfono, etc.
- Servidor Dell R730, de 20 núcleos y 128 GB de RAM.

Existen varias licencias de software gratuitas en internet, como por ejemplo: Julia, Python, Java, R, Latex, etc.

References

Bahmani, B., Moseley, B., Vattani, A., Kumar, R., and Vassilvitskii, S. (2012). Scalable K-Means ++. Proceedings of the VLDB Endowment (PVLDB), 5:622–633.

Balcan, M. F., Ehrlich, S., and Liang, Y. (2013). Distributed k -Means and k -Median Clustering on General Topologies. Advances in Neural Information Processing Systems 26 (NIPS 2013), pages 1–9.

Crestani, F., and Markov, I. (2013). Distributed Information Retrieval and Applications. 35th European Conference on IR Research, 865–868.

Das, A., Datar, M., Garg, A., and Rajaram, S. (2007). Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web, pages 271–280. ACM.

De Vries, C. M., De Vine, L., Geva, S., and Nayak, R. (2015). Parallel streaming signature em-tree: A clustering algorithm for web scale applications. In Proceedings of the 24th International Conference on World Wide Web, pages 216–226. ACM.

Dhillon, I. S. and Modha, D. S. (1999). A data-clustering algorithm on distributed memory multiprocessors. LargeScale Parallel Data Mining, 1759(802):245–260.

Ene, A., Im, S., and Moseley, B. (2011). Fast Clustering using MapReduce. Kdd, 681–689.

Ertöz, L., Steinbach, M., and Kumar, V. (2003). Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. Proceedings of the SIAM International Conference on Data Mining, 47–58.

Forman, G. and Zhang, B. (2000). Distributed data clustering can be efficient and exact. ACM SIGKDD Explorations Newsletter, 2(2):34–38.

Han, J., Pei, J., and Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

Jagannathan, G., and Wright, R. N. (2005). Privacy-preserving Distributed K-means Clustering over Arbitrarily Partitioned Data. Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 593–599.

Januzaj, E., Kriegel, H.-P., and Pfeifle, M. (2003). Towards Effective and Efficient Distributed Clustering. Workshop on Clustering Large Data Sets, pages 49–58.

Januzaj, E., Kriegel, H.-P., and Pfeifle, M. (2004). Scalable Density-Based Distributed Clustering. pages 231–244.

Jin, C., Chen, Z., Hendrix, W., Agrawal, A., and Choudhary, A. (2015). Incremental, Distributed Single-linkage Hierarchical Clustering Algorithm Using Mapreduce. Proceedings of the Symposium on High Performance Computing, pages 83–92.

Johnson, E. and Kargupta, H. (2000). Collective, hierarchical clustering from distributed, heterogeneous data. Lecture Notes in Computer Science, 1759:221–244.

Kargupta, H., Huang, W., Sivakumar, K., and Johnson, E. (2001). Distributed clustering using collective principal component analysis. Knowledge and Information Systems, 3(4):422–448.

Klusch, M., Lodi, S., and Moro, G. (2003). Distributed clustering based on sampling local density estimates. IJCAI International Joint Conference on Artificial Intelligence, pages 485–490.

Kriegel, H.-p., Kr, P., Pryakhin, A., and Schubert, M. (2005). Effective and Efficient Distributed Model-based Clustering.

Li, T., Zhu, S., and Ogihara, M. (2003). Algorithms for Clustering High Dimensional and Distributed Data. Intelligent Data Analysis Journal, 7(February):1–36.

Liang, Y., Balcan, M.-f., and Kanchanapally, V. (2013). Distributed PCA and k-Means Clustering. The Big Learning Workshop in NIPS 2013, pages 1–8.

Liu, J., Huang, J. Z., Luo, J., and Xiong, L. (2012). Privacy Preserving Distributed DBSCAN Clustering. Proceedings of the 2012 Joint EDBT/ICDT Workshops, 177–185.

Merugu, S. and Ghosh, J. (2003). Privacy-preserving Distributed Clustering using Generative Models. Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM), pages 0–7.

Nagwani, N. K. (2015). Summarizing large text collection using topic modeling and clustering based on MapReduce framework. Journal of Big Data, 2:1–18.

Naldi, M. C. and Campello, R. J. G. B. (2014). Evolutionary k-means for distributed data sets. Neurocomputing, 127:30–42.

Qi, Z., Jinze, L., and Wei, W. (2008). Approximate clustering on distributed data streams. Proceedings - International Conference on Data Engineering, 00:1131–1139.

Rousseeuw Peter J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53–65.

Xu, X., Jäger, J., and Kriegel, H. (2002). A fast parallel clustering algorithm for large spatial databases. High Performance Data Mining, 290:263–290.