A Review on Efficiency of K-Means Algorithm with Canopy Clustering in Hadoop

Aniket Gavhale, Punam Pofale, Sayali Baitule, Sushil Samarth
Department of Computer Technology
Rashtrasant Tukadoji Maharaj Nagpur University,Nagpur, India
aniket.gavhale@gmail.com,sushil.samarth@gmail.com,punampofale@gmail.com,sayali_baitule@gmail.com

Abstract— **There are very big bottlenecks when traditional data mining algorithms deal with large data sets. The emergence of Distributed Computing has solved bottlenecks for massive data storage and computing and made massive data mining become possible.**

Many important problems involved in clustering large datasets. There are three different ways in which the dataset can be large: first, there can be a large number of elements in the dataset, second, each element can have many features, and third, there can be many clusters to discover. Here in this papers a novel technique for clustering the large and high dimensional datasets. The main idea is to use an inexpensive and approximate distance measure inorder to efficiently partition the data into overlapping subsets which is called as canopies. After we get these canopies the desired clustering is performed by measuring exact distances only between points that occur in a common canopy. Using canopies, large clustering problems that were formerly impossible become practical and efficient.

In the field of data mining, clustering is one of the important areas. K-Means is a typical distance-based clustering algorithm. Here, the canopy clustering algorithm is implemented as an efficient clustering technique by means of knowledge integration. With the study of the canopy clustering the K-Means paradigm of computing, we find is appropriate for the implementation of a clustering algorithm. This paper shows some advantages of canopy cluster to K-Means clustering mechanism and proposes a pre clustering approach to K-Means Clustering method. The proposed method aims to apply the clustering algorithm effectively to the given dataset. The extensive studies demonstrate that the proposed algorithm is more efficient and the performance is stable. Thus, by improving time performance we can make an efficient clustering technique. Here we use Hadoop's MapReduce program model for Kmeans clustering with canopy clustering.

Index Terms—**Data mining, Clustering, K-Means Clustering, Canopy Clustering, Hadoop, MapReduce program model.**

I. Introduction

Data mining is defined as the middle layer of computer science and statistics. It helps in analyzing and summarizing data into useful information. That information can be used to increase capitals and computation complexities. There are various tools available for analyzing and extraction useful pattern from that. Data mining software is one of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles. It does the categorization and summarization on the relationship identity. There are various methods and techniques in data mining process such as of artificial intelligence, machine learning and statistics.

Data mining involves some common classes of tasks such as anomaly detection, Association rule learning , Clustering, Classification, Regression, Summarization, supervised learning and unsupervised learning algorithms.

Supervised learning deals with the labeled values of the m samples in the training set T are known.

- Supervised learning generates a function that maps inputs to desired outputs (also called labels).
- In a classification problem, the learner approximates a function mapping; a vector into classes by looking at input-output examples of the function.

In short we can tell that supervised learning is the machine learning task of inferring a function from labeled set of training examples called as training data.

Unsupervised learning deals with the training set of vectors without labeled values. The problem in this case, typically, is to partition the training set into subsets, in some appropriate way. Unsupervised learning methods have application in taxonomic problems in which it is desired to invent ways to classify data into meaningful categories. Example of unsupervised learning is clustering mechanism.

In short we can tell that unsupervised learning algorithms operate on unlabelled examples, i.e., input where the desired output is indefinite. Here the goal is to discover structure in the data from inputs to outputs.

Consider the various sets of points in a two-dimensional space illustrated in the figure1 below. The first set (1.a) Seems naturally partition able into two classes, while the second (1.b) Seems difficult to partition at all, and the third (1.c) Is problematic.

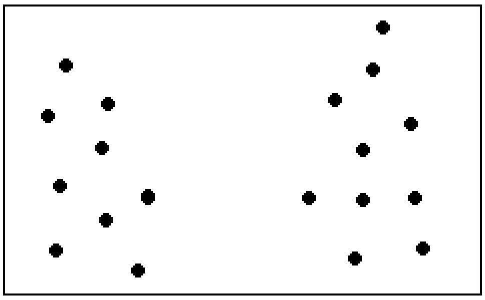

Fig. 1.a: two clusters

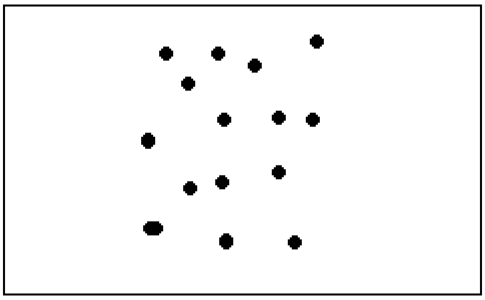

Fig. 1.b: one cluster

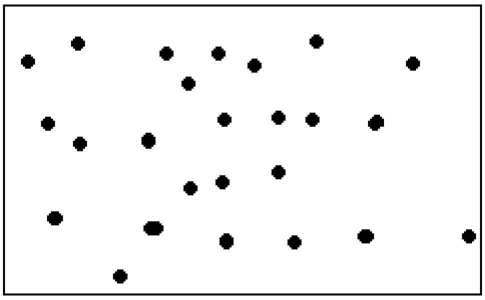

Fig. 1.c: Problematic?
Fig1: Patterns in a two-dimensional

Unsupervised learning attempts to find natural partitions of patterns. There are two stages involved:

Stage1: Form an R-way partition of a set of unlabeled training patterns. The partition separates into R mutually exclusive and extensive subsets, called as clusters.

Stage2: Design a classifier based on the labels assigned to the training patterns by the partition.

The key idea of applying new mechanism of canopy clustering is to perform clustering in two stages, first a rough and quick stage that divides the data into overlapping subsets we call “canopies” then added rigorous final stage in which expensive distance measurements are only made among points that occur in a common canopy. This differs from previous clustering methods because it uses two different distance metrics for the two stages, and forms overlapping regions. The first stage can make use of extremely inexpensive methods for finding data elements near the center of a region. Many proximity measurement methods, such as the inverted index commonly used in information retrieval systems, are very efficient in high dimensions and can find elements near the query by examining only a small fraction of a data set. Variants of the inverted index can also work for real-valued data.

Once the canopies are built using the approximate distance measure, the second stage completes the clustering by running a standard clustering algorithm using a rigorous distance metric. However, significant computation is saved by eliminating all of the distance comparisons among points that do not fall within a common canopy. Clustering based on canopies can be applied to many different fundamental clustering algorithms.

In this study we explore a novel concept know as canopy clustering where data is first clustered into overlapping canopies using a comparatively cheap distance measure, then the data is clustered further using the expensive aforementioned clustering algorithms. Canopy clustering has been proven not to reduce the clustering accuracy but instead improve computational efficiency.

Hadoop was created by Doug Cutting; he is person behind the Apache Lucene creation, Apache Luence is the text search library which is being widely used. Hadoop has origin in Apache Nutch, Apache Nutch is an open source search engine and it is a web search engine, which is a part of the Lucene project. Apache Hadoop is a software framework that supports data-intensive distributed applications. It empowers the applications to work with thousands of computational autonomous and independent computers and petabytes of data. Hadoop is the derivative of Google's MapReduce and Google File System (GFS) . These include reliability achieved by replication, scales well to thousands of nodes, can handle petabytes of data, automatic handling of node failures, and is designed to run well on heterogeneous commodity class hardwares. However, Hadoop is still a fairly new project and limited example code and documentation is available for non-trivial applications.

II. BACKGROUND STUDY

The original k-means algorithm is very impressionable to the initial starting points. So, it is quite crucial for k-means to have refine initial cluster centers. Several methods have been proposed in the literature for finding the better initial centroids. And some methods were proposed to improve both the accuracy and efficiency of the k-means clustering algorithm. In this paper, some of the more recent proposals are reviewed.

A. M. Fahim proposed an enhanced method for assigning data points to the suitable clusters. In the original kmeans algorithm in each iteration the distance is calculated between each data element to all centroids and the required computational time of this algorithm is depends on the number of data elements, number of clusters and number of iterations, so it is computationally expensive. In Fahim approach the required computational time is reduced when assigning the data elements to the appropriate clusters. But in this method the initial centroids are selected randomly. So this method is very sensitive to the initial starting points and it does not promise to produce the unique clustering results.

K. A. Abdul Nazeer proposed an enhanced algorithm to improve the accuracy and efficiency of the kmeans clustering algorithm. In this algorithm two methods are used, one method for finding the better initial centroids. And another method for an efficient way for assigning data points to appropriate clusters with reduced time complexity. This algorithm produces good clusters in less amount of computational time.

Mahesh Maurya and Sunita Mahajan demonstrate to analyze the performance of MapReduce program model on Hadoop cluster. In their Experiment, Performance of MapReduce application has been shown with respect to execution time and number of nodes. They find that as the number of nodes increases the execution time decreases.

Jing Zhang and Xindong Wu et al takes advantages of KMeans, 2-tier clustering mechanism and Map-Reduce computing model; proposes a new method for parallel and distributed clustering to explore distributed clustering problem based on Map-Reduce. The method aims to apply the clustering algorithm effectively to the distributed environment. The extensive studies demonstrate that the proposed algorithm is scalable, and the time performance is stable. Meanwhile, adding number of cluster nodes would improve the time performance of clustering.

A. MapReduce

MapReduce is a programming model for expressing distributed computations on massive amounts of data, and also an execution framework for large-scale data processing on clusters of commodity servers. In other word it can tell that MapReduce represents to a framework that runs on a computational cluster to extract the Knowledge from a large datasets. The name MapReduce is derived from two functions map ( ) and reduce ( ) functions. The Map ( ) function usually applies to all the members of the dataset and then returns a list of results. And “Reduce ( ) function” collates and resolves the results from one or more mapping operations executed in parallel.

MapReduce Model splits the input dataset into independent chunks called as subsets, which are processed by map ( ) and reduce ( ). Generally, compute nodes & storage nodes are the same. That is the entire computation involving map ( ) and reduce ( ) functions will be happening on DataNodes and result of computation is going to be stored locally.

In the MapReduce Model, programs written in the functional style are automatically parallelized and executed on the large cluster of commodity hardware. The run-time system takes care of the details of broken input data, and schedules the program's execution across the number of machines; it handles the machine failures, and manages inter-machine communication. In this way without any experience with parallel and distributed systems this allows programmers, to easily utilize the resources of a large distributed system

1) MapReduce Design : In MapReduce, records are treated in isolation by tasks called as Mappers. The output from the Mappers is then brought together into a second set of tasks called as Reducers; here results from many different mappers are being merged together. Problems suitable for processing with MapReduce must usually be easily split into independent subtasks that can be processed in parallel. The map and reduce functions are both specified in terms of data is structured in key-value pairs.

$$\text{map: } (k_1, v_1) \rightarrow [(k_2, v_2)]$$
$$\text{reduce: } (k_2, [v_2]) \rightarrow [(k_3, v_3)]$$

The power of MapReduce is from the execution of many map tasks which run in parallel on a data set and gives output of the processed data in form of intermediate key-value pairs. Each reduce task receives and processes data for one particular key at a time and outputs the data which processes as key-value pairs.

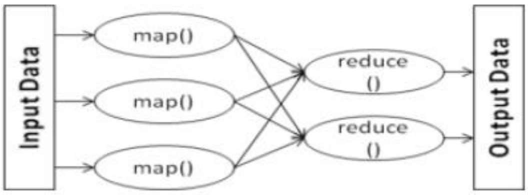

Fig 2: MapReduce key-value pair’s generation

2) MapReduce Strategy : The Map invocations are distributed across multiple machines by automatically partitioning the input data into a set of M chunks. The input chunks can be processed in parallel by different machines. Reduce requests are being distributed by partitioning the intermediate key space into R pieces using a partitioning function (e.g., hash (key) mod R). The number of partitions (R) and the partitioning function are specified by the user. Figure give below shows the overall flow of a MapReduce operation. As soon as the MapReduce function is called by the user program, the following sequence starts

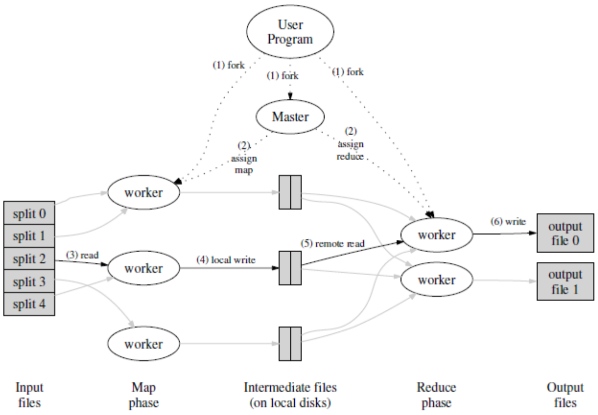

Fig 3: MapReduce Execution overview

B. Hadoop

Apache Hadoop is an open source it is built on Java framework and it is built for implementing the reliable and scalable computational networks which supports data intensive distributed applications, and it is licensed under Apache v2 license. It enables many applications to work with thousands and thousands of computational independent computers and petabytes of the data. Hadoop was derived from Google's MapReduce and Google File System (GFS). Hadoop is a top-level Apache project which is built and used by a global communal of contributors, it has been written in the Java programming language. Hadoop includes several subprojects: HDFS, MapReduce, HBase, Pig, Hive, ZooKeeper.

C. HDFS

The Hadoop Distributed File System (HDFS) is designed to store large data sets with high reliability, and to stream those data sets with high bandwidth. In a large cluster, more that thousands of servers both host and Client are directly attached to storage and execute user application tasks. Hadoop provides a distributed file system and a framework for the analysis and transformation of very large data sets using the MapReduce paradigm.

HDFS is the file system constituent the Hadoop. While the interface to HDFS is being formed after the UNIX file system, the truthfulness to standards was sacrificed in favor of enhanced performance for the applications at hand. HDFS stores the file system, metadata and the application data independently. Alike to distributed file systems, like PVFS, Lustre and GFS. HDFS stores metadata on a dedicated server, known as NameNode. Application data are stored on other servers known as DataNodes. All servers are connected and communicate with each other using TCP-based protocols.

III. RELATED WORK

A. Clustering Algorithms

Data clustering is the partitioning of a data set or sets of data into similar subsets; this can be accomplished by using some of the clustering algorithms.

Clustering algorithms can be categorized based on their cluster model, as listed above. The following overview will only list the most prominent examples of clustering algorithms. Clustering algorithms can be categorized as connectivity based clustering (hierarchical clustering), centroid-based clustering, distribution-based clustering, density-based clustering.

1) *Connectivity based clustering (Hierarchical clustering)* Connectivity based clustering, also known as hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away. As such, these algorithms connect "objects" to form "clusters" based on their distance. Single-linkage clustering and complete linkage clustering are the some example of this method.

2) *Centroid-based clustering* : In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the data set. Here we find the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized. Lloyd's algorithm, often actually referred to as "k-means algorithm" is the example of this method.

3) *Distribution-based clustering* : The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution. A nice property of this approach is that this closely resembles the way artificial data sets are generated by sampling the random objects from the given distribution. Expectation-maximization algorithm is the example of this method.

4) *Density-based clustering* : In density-based clustering, clusters are defined as areas of higher density than the remainder of the data set. Objects in these sparse areas - that are required to separate clusters - are usually considered to be noise and border points. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is the example of this method.

B. K-Means Clustering Algorithm

K-MEANS is the simplest algorithm used for clustering also it an unsupervised clustering algorithm. The K-Means algorithm is used to partitions the data set into k clusters using the cluster mean value so that in the resulting clusters is having high intra cluster similarity and low inter cluster similarity. K-Means clustering algorithm is iterative in nature.

The K-means clustering algorithm is known to be efficient in clustering large data sets. This clustering algorithm was originally developed by MacQueen , and is one of the simplest and the best known unsupervised learning algorithms that solve the well-known clustering problem. The K-Means algorithm targets to partition a set of given objects into k clusters based on their features, where k is a user-defined constant. The core idea is to define k centroids, one centroid for each cluster. The centroid for a cluster is calculated and formed in such a way that it is closely related (in terms of similarity function; similarity can be measured by using different methods such as cosine similarity, Euclidean distance, Extended Jaccard) to all objects in that cluster.

Steps, below explains the working K-MEANS algorithm.

1. Generate “k” number of clusters randomly.
2. Calculate the distance between each data points to each of the centers; each data point is assigned to a cluster.
3. The centroids are updated by calculating the mean value of all data points in the respective cluster.
4. Repeat step 2, with respect to new centers.
5. If the assignment of cluster for the data points changes, repeat step 3 else stop the process

Although the K-Means algorithm is relatively straight-forward and it is computationally expensive. The total run time of K-Means algorithm O (k*n*i) where,

k= Number of k-centers
n= Number of total points

i=iterations taken to converge

Thus, the current research must work on for reducing the run time and hence improving the efficiency of the algorithm.

C. *Canopy Clustering Algorithm*

Canopy Clustering an unsupervised pre-clustering algorithm related to the K-means algorithm, which can process huge data sets efficiently, but the resulting "clusters" are merely a rough pre-partitioning of the data set to then analyze the partitions with existing slower methods such as k-means clustering.

The basic steps of the canopy clustering are described below :

Given two threshold distance T1 and T2; T1>T2 and a collection of points. Now, to determine the Canopy Centers: there is iteration through the set of points, if the point is at distance decide the canopy membership - for each point in the input set if the point is at a distance < T1 from any of points in the list of canopy centers (generated in step) then point is member of the corresponding cluster.

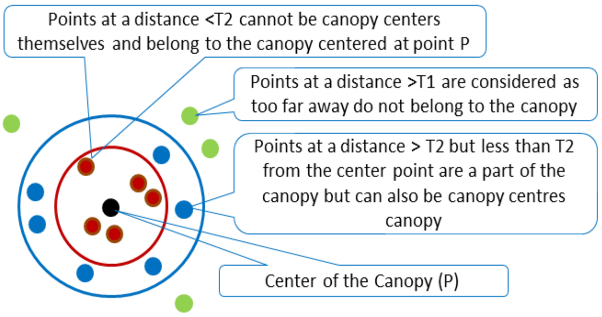
Points at a distance <T2 cannot be canopy centers themselves and belong to the canopy centered at point P
Points at a distance >T1 are considered as too far away do not belong to the canopy
Points at a distance > T2 but less than T2 from the center point are a part of the canopy but can also be canopy centres canopy
Center of the Canopy (P)

Fig 4: Canopy clustering description

D. *K-Means Clustering algorithm with Canopy Clustering*

K-Means algorithm when used with canopy clustering can reduce the computations in the distance calculation step hence improving the efficiency of the algorithm.

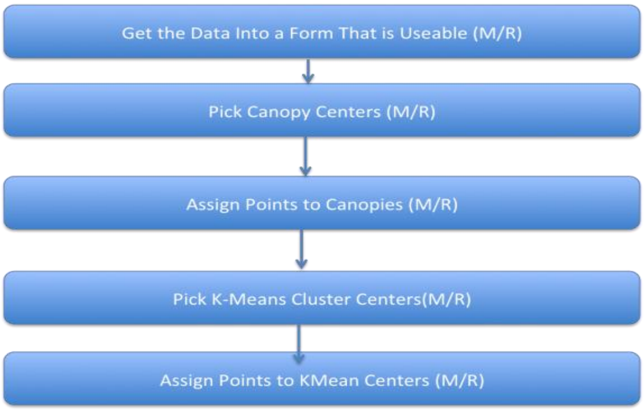
Get the Data Into a Form That is Useable (M/R)
Pick Canopy Centers (M/R)
Assign Points to Canopies (M/R)
Pick K-Means Cluster Centers(M/R)
Assign Points to KMean Centers (M/R)

Fig 5: MapReduce Steps of K-Means algorithm with Canopy Clustering

The Map Reduce implementation of K-Means Algorithm with Canopy Clustering has the following steps.

1. Data points Preparation: The input data needs to be converted into a format suitable for distance and similarity measures. This is trivial for integer data but requires some preprocessing of text data. The details of the pre-processing are discussed later.
2. Picking Canopy Centers
3. Assign data points to canopy centers: The canopy assignment step would simply assign data points to generated canopy centers.
4. Pick K-Mean Cluster Centers & Iterate until convergence. The computation to calculate the closest k-mean center is greatly reduced as we only calculate the distance between a k-center and data point if they share a canopy. The average emitted is the new k-center. We repeat the iteration until the centers don't move much between iterations.
5. Assign data points to the K-Mean Centers: The final step of the algorithm assigns data points to the final k mean centers. The data points are now in clustered sets.

IV. REVIEW

A. *K-Means V. K-Means with Canopy*

General approach to k-means clustering

1. Collect: Any method.
2. Prepare: Numeric values are needed for a calculation of distance, and nominal values can be mapped into binary values for distance calculations.
3. Analyze: Any method.
4. Train: Doesn't apply to unsupervised learning.
5. Test: Apply the clustering algorithm and do results inspection.. Quantitative error measurements such as sum of squared error (introduced later) can be used.
6. Use: Anything you wish. Frequently, the clusters centers can be treated as representative data of the whole cluster to make decisions.

General approach to k-means clustering using Canopy clustering

1. Simple distance calculation will be used for Canopy clustering.
2. Expensive distance calculation will be used for K-means clustering.
3. Output of Canopy cluster will become input of K-means clustering.
4. Apply Cosine similarity metric to find out similar users.

The algorithm proceeds as follows:

5. Cheaply partitioning the data into overlapping subsets called "canopies".

6. Perform more expensive clustering only within these canopies

Canopy Clustering is often used as an initial step in K-Means Clustering which is more rigorous clustering techniques. By starting with an initial clustering the number of more expensive distance measurements can be significantly reduced by ignoring points outside of the initial canopies.

B. Why hadoop is choose ?

The reason behind the selection of Hadoop for Clustering is, following advantageous feature of Hadoop

Distribute data and computation.The computation local to data prevents the network overload.
Tasks are independent The task are independent so,

a. We can easy to handle partial failure. Here the entire nodes can fail and restart.

b. it avoids crawling horrors of failure and tolerant synchronous distributed systems.

c. Speculative execution to work around stragglers.

Linear scaling in the ideal case.It used to design for cheap, commodity hardware.

HDFS store large amount of information. HDFS is simple and robust coherency model. That is it should store data reliably. HDFS is scalable and fast access to this information and it also possible to serve s large number of clients by simply adding more machines to the cluster. HDFS should integrate well with Hadoop MapReduce, allowing data to be read and computed upon locally when possible. HDFS provide streaming read performance. Data will be written to the HDFS once and then read several times.

V. CONCLUSION

The canopy clustering algorithm is also an unsupervised pre-clustering algorithm, often used as preprocessing step for K-means algorithm or Hierarchical clustering algorithm. It intended to speed up the clustering operations on large data sets.

Since the algorithm uses distance functions and requires the specification of distance thresholds, its applicability for high-dimensional data is limited by the curse of dimensionality. Only when a cheap and approximate – low-dimensional – distance function is available, the produced canopies will preserve the clusters produced by K-means.

The new method has reduced the comparison of the number of instances at each step and there is some evidence that the resulting clusters are improved.

Canopy clustering is a very simple, fast and surprisingly accurate method for grouping objects into clusters, thus it can be used in MapReduce concept using hadoop cluster in order to enhance the clustering techniques. One can also use the canopies idea to speed up prototype based clustering methods like K-means and Expectation-Maximization (EM).

REFERENCES

[1] Mahesh Maurya, Sunita Mahajan, “Performance analysis of MapReduce Programs on Hadoop cluster”, IEEE.

[2] Jing Zhang, Xindong Wu “A 2-Tier Clustering Algorithm with Map-Reduce”, IEEE 2010.

[3] McCallum, A.; Nigam, K.; and Ungar L.H. (2000) "Efficient Clustering of High Dimensional Data Sets with Application to Reference Matching", Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining.

[4] A. M. Fahim, A. M. Salem, F. A. Torkey and M. A. Ramadan, “An Efficient enhanced k-means clustering algorithm,” journal of Zhejiang

[5] K. A. Abdul Nazeer and M. P. Sebastian, “Improving the accuracy and efficiency of the k-means clustering algorithm,” in International Conference on Data Mining and Knowledge Engineering (ICDMKE), Proceedings of the World Congress on Engineering (WCE-2009),

[6] Apache Hadoop. http://hadoop.apache.org/

[7] http://mahout.apache.org/users/clustering/canopy-clustering.html

[8] http://en.wikipedia.org/wiki/Canopy_clustering_algorithm

[9] Tom White, “Hadoop: The Definitive Guide”, 2009 Published by O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472