The Journal of
Mathematics and Computer Science

Available online at
http://www.TJMCS.com

The Journal of Mathematics and Computer Science Vol .5 No.3 (2012) 229-240

A survey of hierarchical clustering algorithms

Marjan Kuchaki Rafsanjani
Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran
kuchaki@uk.ac.ir

Zahra Asghari Varzaneh
Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran
asghari_za@yahoo.com

Nasibeh Emami Chukanlo
Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran
nasibeh.emami@yahoo.com

Received: November 2012, Revised: December 2012
Online Publication: December 2012

Abstract

Clustering algorithms classify data points into meaningful groups based on their similarity to exploit useful information from data points. They can be divided into categories: Sequential algorithms, Hierarchical clustering algorithms, Clustering algorithms based on cost function optimization and others. In this paper, we discuss some hierarchical clustering algorithms and their attributes, and then compare them with each other.

Keywords: Clustering; Hierarchical clustering algorithms; Complexity.

2010 Mathematics Subject Classification: Primary 91C20; Secondary 62D05.

1. Introduction.

Clustering is the unsupervised classification of data into groups/clusters [1]. It is widely used in biological and medical applications, computer vision, robotics, geographical data, and so on [2]. To date, many clustering algorithms have been developed. They can organize a data set into a number of groups /clusters. Clustering algorithms may be divided into the following major categories: Sequential algorithms, Hierarchical clustering algorithms, Clustering algorithms based on cost function optimization and etc [3]. In this paper, we only focus on hierarchical clustering algorithms. At first, we introduce some hierarchical clustering algorithms then we compare them.

The paper is organized as follows: Section 2 describes clustering process. Section 3 denotes categories of clustering algorithms. Section 4 describes Hierarchical clustering algorithms. Section 5 explains specific hierarchical clustering algorithms. Comparison between algorithms present in section 6 and finally paper conclusion and future work are provided in section 7.

2. Clustering process

Clustering is the unsupervised classification of data into groups/clusters [1]. The input for a system of cluster analysis is a set of samples and a measure of similarity (or dissimilarity) between two samples. The output from cluster analysis is a number of groups /clusters that form a partition, or a structure of partitions, of the data set (Fig. 1). The ultimate goal of clustering can be mathematically described as follows:

$$X = C_1 \cup \cdots C_i \cup C_n; \quad C_i \cap C_j = \phi \ (i \neq j), \qquad (1)$$

Where X denotes the original data set, C_i, C_j are clusters of X, and n is the number of clusters [5].

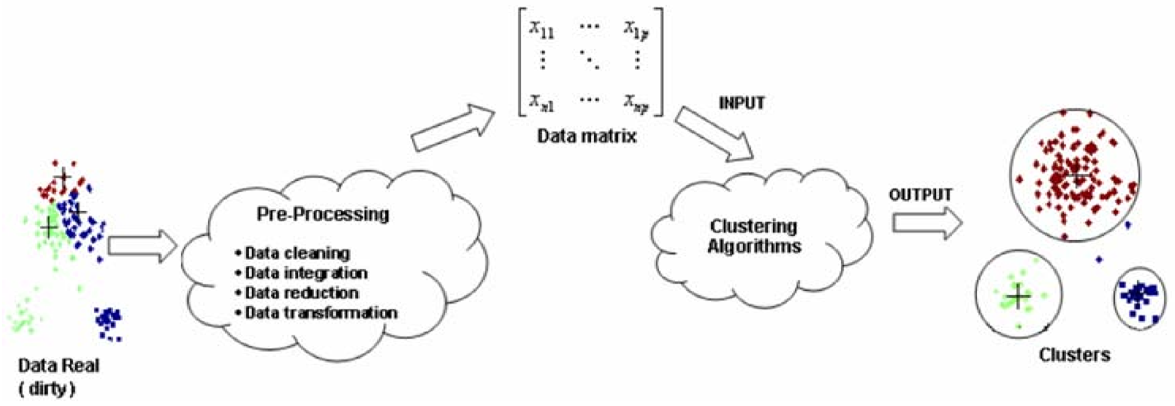


Fig. 1. Clustering process [5]

Data Pre-processing describes any type of processing performed on raw data to prepare it for another processing procedure. It's product is training set and including: Data cleaning, which fill in missing values, smooth noisy data, identify or remove outliers, and resolve in consistencies; Data integration, which integration of multiple data bases, data cubes, or files; Data transformation, which is normalization and aggregation; Data reduction, which obtains reduced representation in volume but produces the same or similar analytical results [5].

3. Clustering algorithms

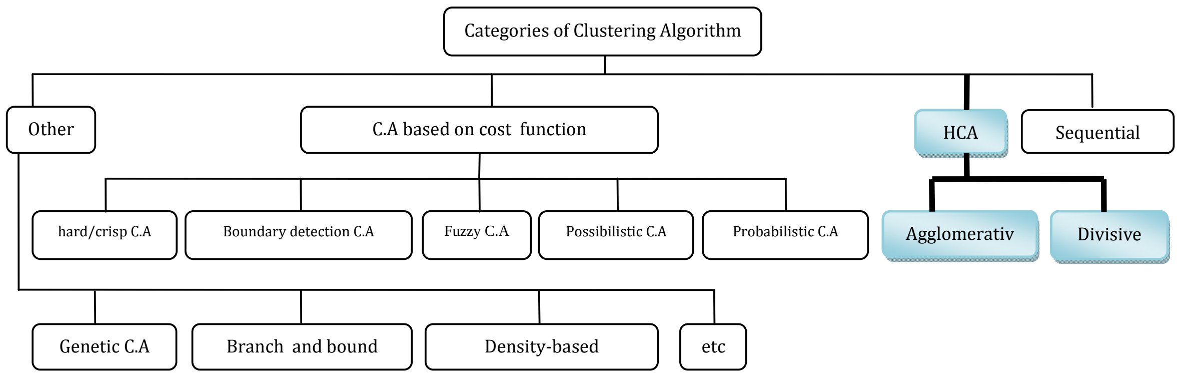
In this section clustering algorithms divided into the following major categories [3]:

– *Sequential algorithms:* These algorithms produce a single clustering. They are quite straightforward and fast methods. In most of them, all the feature vectors are presented to the algorithm once or a few times (typically no more than five or six times). The final result is, usually, dependent on the order in which the vectors are presented to the algorithm.
– *Hierarchical clustering algorithms:* These schemes are further divided into

- *Agglomerative algorithms (bottom-up, merging):* These algorithms produce a sequence of clustering of decreasing number of clusters, *m*, at each step. The clustering produced at each step results from the previous one by merging two clusters into one.
- *Divisive algorithms (top-down, splitting):* These algorithms act in the opposite direction; that is, they produce a sequence of clustering of increasing *m* at each step. The clustering produced at each step results from the previous one by splitting a single cluster into two.

– *Clustering algorithms based on cost function optimization:* This category contains algorithms in which "sensible" is quantified by a cost function, *J*, in terms of which a clustering is evaluated. Usually, the number of clusters *m* is kept fixed. Most of these algorithms use differential calculus concepts a produce successive clustering while trying to optimize *J.* Algorithms of this category are also called iterative function optimization schemes. This category includes: Hard or crisp clustering algorithms, Probabilistic clustering algorithms, Fuzzy clustering algorithms, Possibilistic clustering algorithms and Boundary detection algorithms.

– *Other*: This last category contains some special clustering techniques that do not fit nicely in any of the previous categories. These include: Branch and bound clustering algorithms, Genetic clustering algorithms, stochastic relaxation methods, Density-based algorithms and etc. Fig. 2 illustrates this category.



C.A: Clustering Algorithm, HCA: Hierarchical clustering algorithms

Fig. 2. Clustering category

4. Hierarchical clustering algorithms

The hierarchical methods group training data into a tree of clusters. This tree structure called dendrogram (Fig. 3). It represents a sequence of nested cluster which constructed top-down or bottom-up. The root of the tree represents one cluster, containing all data points, while at the leaves of the tree, there are n clusters, each containing one data point. By cutting the tree at a desired level, a clustering of the data points into disjoint groups is obtained [6].

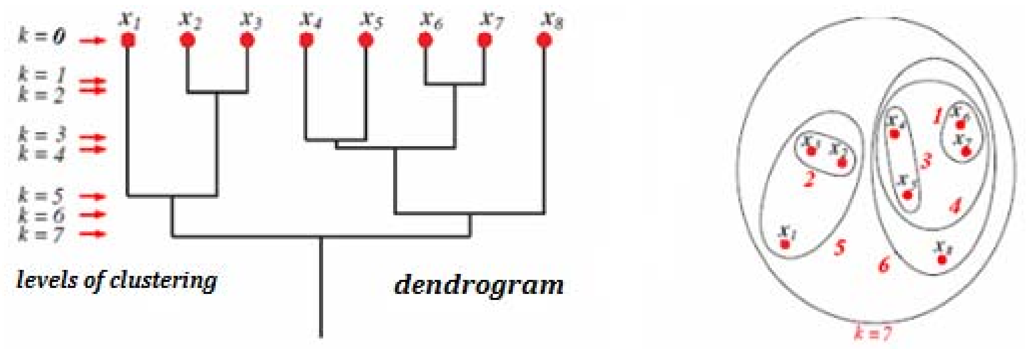


Fig. 3. Tree structure of training data (dendrogram)[5].

Hierarchical clustering algorithms divide into two categories: Agglomerative and Divisive. Agglomerative clustering executes in a bottom–top fashion, which initially treats each data point as a singleton cluster and then successively merges clusters until all points have been merged into a single remaining cluster. Divisive clustering, on the other hand, initially treats all the data points in one cluster and then split them gradually until the desired number of clusters is obtained. To be specific, two major steps are in order. The first one is to choose a suitable cluster to split and the second one is to determine how to split the selected cluster into two new clusters [7]. Fig 4 shows structure of clustering algorithms. Many agglomerative clustering algorithms have been proposed, such as CURE, ROCK, CHAMELEON, BIRCH, single-link, complete-link, average-link, Leaders-Subleaders. One representative divisive clustering algorithm is the bisecting k-means method. Fig 5 is a representation of Hierarchical clustering schemes.

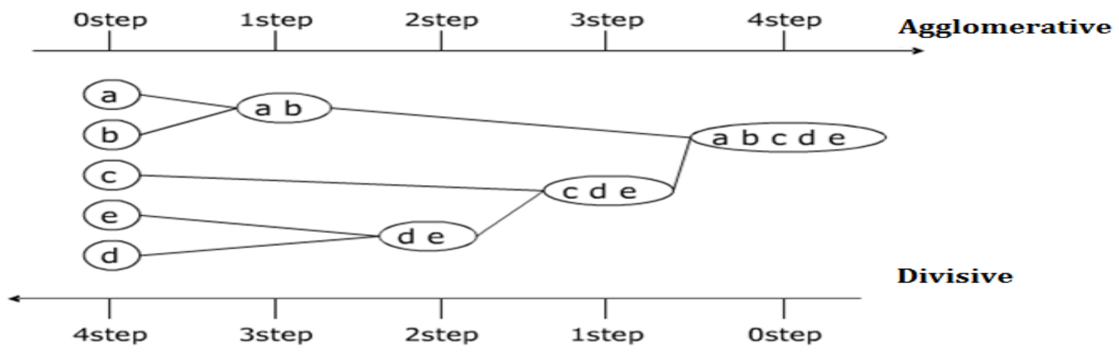


Fig. 4. Application of agglomerative and divisive to a data set of five objects, {a, b, c, d, e}[5].

Agglomerative and Divisive clustering have been applied to document clustering [8]. Divisive clustering is very useful in linguistics, information retrieval, and document clustering applications [7].

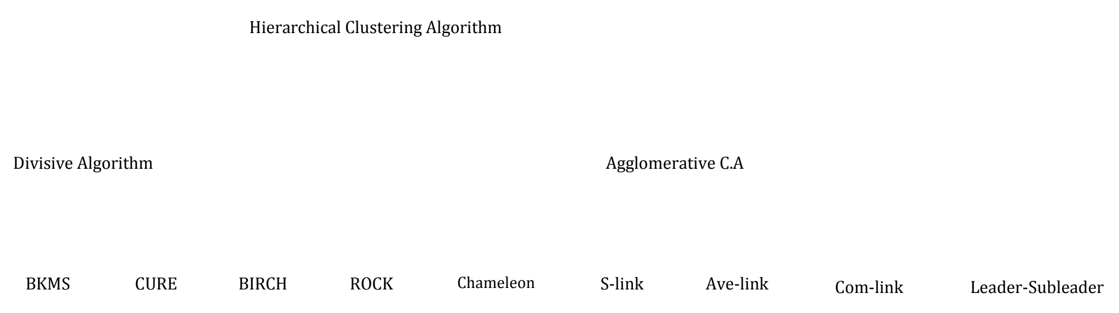


Fig. 5. Hierarchical Clustering Algorithm

5. Specific algorithms

We focus on hierarchical clustering algorithms. At first, we introduce these algorithms in subsections and then compare them by different criteria.

5.l. CURE (Clustering Using REpresentatives)

In this section, we present CURE's agglomerative hierarchical clustering algorithm. It first partitions the random sample and partially clusters the data points in each partition. After eliminating outliers, the pre clustered data in each partition is then clustered in a final pass to generate the final clusters. Fig 6 is an overview of CURE. CURE algorithm salient features are: (1) the clustering algorithm can recognize arbitrarily shaped clusters (e.g., ellipsoidal), (2) the algorithm is robust to the presence of outliers, and (3) the algorithm uses space that is linear in the input size n and has a worst-case time complexity of O(n^2 logn). For lower dimensions (e.g., two), the complexity can be shown to further reduce to O(n^2). (4) It appropriate for handling large data sets [9].

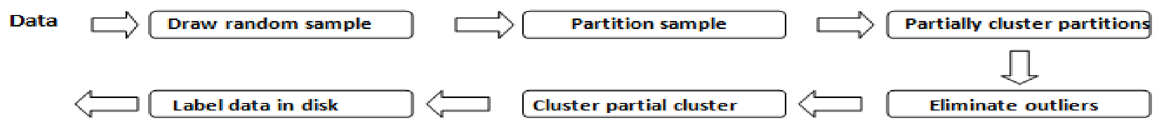


Fig. 6. CURE process [9]

There is a developed algorithm denoted as CURE that combines centroid and single linkage approaches by choosing more than one representative point from each cluster. At each step of the algorithm, the two clusters with the closest pair of representative points (one in each cluster) are merged [10].

5.1.1 Disadvantage of CURE

– CURE ignores the information about the aggregate inter-connectivity of objects in two clusters. So it is introduced Chameleon algorithm [5].

5.2 BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)

BIRCH is an agglomerative hierarchical clustering algorithm proposed by Charikar et al. in 1997[11]. It is especially suitable for very large databases. This method has been designed so as to minimize the number of I/O operations [3]. BIRCH incrementally and dynamically clusters incoming multi-dimensional metric data points to try to produce the best quality clustering with the available resources (i. e., available memory and time constraints). BIRCH can typically find a good clustering with a single scan of the data, and improve the quality further with a few additional scans. BIRCH is also the first clustering algorithm proposed in the database area to handle ''noise'' (data points that are not part of the underlying pattern) effectively. Fig 7 presents the overview of BIRCH [12].

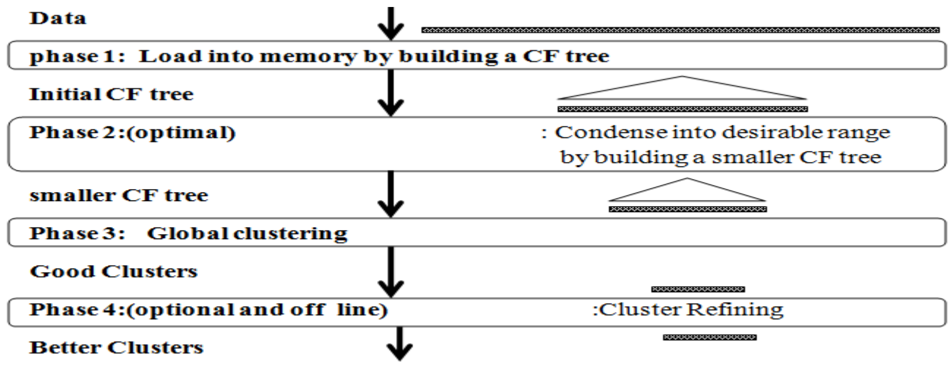

Data
phase 1: Load into memory by building a CF tree
Initial CF tree
Phase 2:(optimal)
: Condense into desirable range by building a smaller CF tree
smaller CF tree
Phase 3: Global clustering
Good Clusters
Phase 4:(optional and off line)
:Cluster Refining
Better Clusters

Fig. 7. BIRCH process [12]

The data pre-processing algorithm BIRCH groups the data set into compact sub clusters that have summary statistics (called Clustering Features (CF)) associated to each of them. These CF's are computed and updated as the sub clusters are being constructed. The end result is an ``in-memory'' summary of the data, where ``local'' compact sub clusters are represented by appropriate summary statistics [13]. BIRCH can achieve a computational complexity of $O(n)$. Two generalizations of BIRCH, known as BUBBLE and BUBBLE-FM algorithms [3].this algorithm can find approximate solution to combinatorial problems with very large data sets [13].

5.2.1 Advantages of BIRCH

- Scales linearly: finds a good clustering with a single scan and improves the quality with a few additional scans.
- Computation complexity of the algorithm is O(n), where n is number-objects.

5.2.2 Disadvantages of BIRCH

- Handles only numeric data, and sensitive to the order of the data record.

– Favors only clusters with spherical shape and similar sizes, because it uses the notion of diameter to control the boundary of a cluster [5].

5.3 ROCK (RObust Clustering using linKs)

ROCK a robust hierarchical-clustering algorithm is an agglomerative hierarchical clustering based on the notion of links [14]. It is appropriate for handling large data sets [3]. ROCK combines, from a conceptual point of view, nearest neighbor, relocation, and hierarchical agglomerative methods. In this algorithm, cluster similarity is based on the number of points from different clusters that have neighbors in common [10].

The steps involved in clustering using ROCK are described in Fig 8. The space complexity of the algorithm depends on the initial size of the local heaps. Therefore space complexity of ROCK's clustering algorithm is $O(\min\{n^2, nm_m m_a\})$, where n is number of input points, m_a and m_m are the average and maximum number of neighbors for a point, respectively. It has a worst-case time complexity of $O(n^2 + nm_m m_a + n^2 \log n)$. A robust hierarchical clustering algorithm ROCK was develop that employs links and not distances for merging clusters [15].

A quick version of the ROCK algorithm for clustering of categorical data is proposed, it is called QROCK. It has the complexity $O(n^2)$. The performance analyses also demonstrate that QROCK is quicker than ROCK [14].

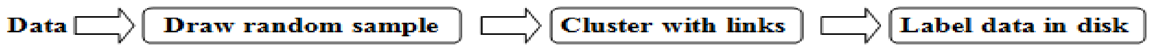


Fig. 8. ROCK process [15]

5.4 CHAMELEON

CHAMELEON as a hierarchical agglomerative clustering algorithm can find dynamic modeling. It is based on two phases: at first partitions the data points into sub-clusters, using a graph partitioning, then repeatedly merging sub-clusters, com from previous stage to obtain final clusters. The algorithm is proven to find clusters of diverse shapes, densities, and sizes in two-dimensional space [17]. Chameleon is an efficient algorithm that uses a dynamic model to obtain clusters of arbitrary shapes and arbitrary densities [2]. Fig 9 provides an overview of the overall approach used by Chameleon to find the clusters in a data point [16].

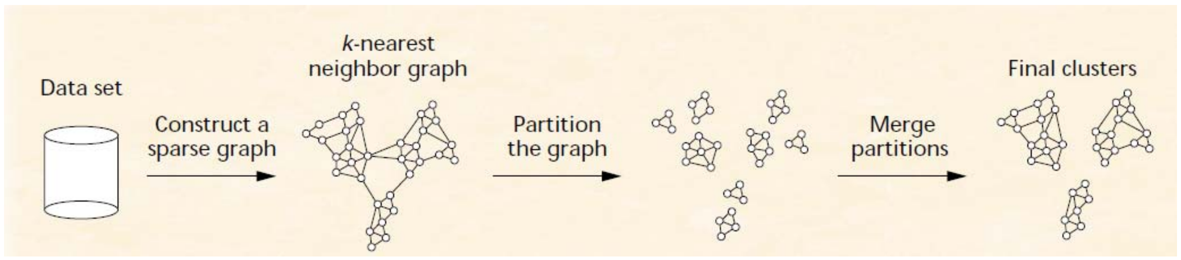


Fig. 9. CHAMELEON process [16]

The algorithm is well suited for the applications where the volume of the available data is large. For large *n*, the worst-case time complexity of the algorithm is $O(n(\log_2 n + m))$, where *m* is the number of clusters formed after completion of the first phase of the algorithm[3].

5.4.1 Disadvantage of CHAMELEON

- CHAMELEON is known for low dimensional spaces, and was not applied to high dimensions. Time complexity of CHAMELEON algorithm in high dimensions is $O(n^2)$[2, 5].

5.5 Linkage algorithms

Linkage algorithms are hierarchical methods that merging of clusters is based on distance between clusters. Three important type of these algorithms are **S**ingle-link(S-link), **Ave**rage-link (Ave-link) and Complete-link (Com-link).They are agglomerative hierarchical algorithms too. The Single-link distance between two subsets is the shortest distance between them, Average-link the average distance and the Complete-link the largest distance [17].
From [1] Single-link time complexity and space complexity is $O(n^2\log n)$ and $O(n^2)$ respectively. The obvious algorithm for computing the Complete-link clustering takes cubic time. Day and Edelsbrunner showed that it can be reduced to $O(n^2\log n)$ time by Using priority queues. Murtagh proposed a quadratic-time algorithm. Later, a quadratic-time algorithm based on the (a, b)-tree data structure was developed by KArivBanek. A quadratic-time algorithm that uses linear space was proposed by Defays; A Parallel implementation of Complete-link clustering have also been developed, but asymptotically the total work was still at least quadratic.Krznaric and Levcopoulos showed that for n points in the Euclidean plane, the Complete-link clustering can be computed in $O(n\log^2 n)$ time and linear space. In addition, they developed an $O(n \log n + n\log^2(1/\varepsilon))$ time algorithm for constructing a Complete-link ε-approximation that uses $O(n)$ space[18].

5.5.1 Disadvantages of linkage algorithm

- **S**ingle-link is sensitive to the presence of outliers and the difficulty in dealing with severe differences in the density of clusters. On the other hand, displays total insensibility to shape and size of clusters [10].
- Average-linkage is sensitive to the shape and size of clusters. Thus, it can easily fail when clusters have complicated forms departing from the hyper spherical shape [10].
- Complete-linkage is not strongly affected by outliers, but can break large clusters, and has trouble with convex shapes [10].

5.6 Leaders–Subleaders

An efficient hierarchical clustering algorithm, suitable for large data sets is proposed by Vijaya, Narasimha Murty and Subramanian . It uses incremental clustering principles to generate a hierarchical structure for finding the subgroups/sub clusters within each cluster. It is called Leaders–Subleaders. Leaders–Subleaders is an extension of the leader algorithm (an incremental algorithm in which L leaders each representing a cluster are generated using a suitable threshold value.). Two major features of Leaders–Subleaders are: effective clustering and prototype selection for pattern classification.

In this algorithm, after finding L leaders using the leader algorithm, subleaders (representatives of the sub clusters) are generated within each cluster represented by a leader, choosing a suitable sub threshold value (Fig 10). Subleaders in turn help in classifying the given new/test data more accurately. This procedure may be extended to more than two levels. An h level hierarchical structure can be generated in only h database scans and is computationally less expensive compared to other hierarchical clustering algorithms. Leaders–Subleaders algorithm require two database scans (h = 2) and its time complexity is O(ndh)(n is number of data point, d is dimensionality of the pattern.) and is computationally less expensive compared to most of the other partitional and hierarchical clustering algorithms. For a h level hierarchical structure, the space complexity is $O((\sum_{j=1}^{h}(L\text{j}))d)$, for h=2 space complexity is $O((L+SL)d)$[19].

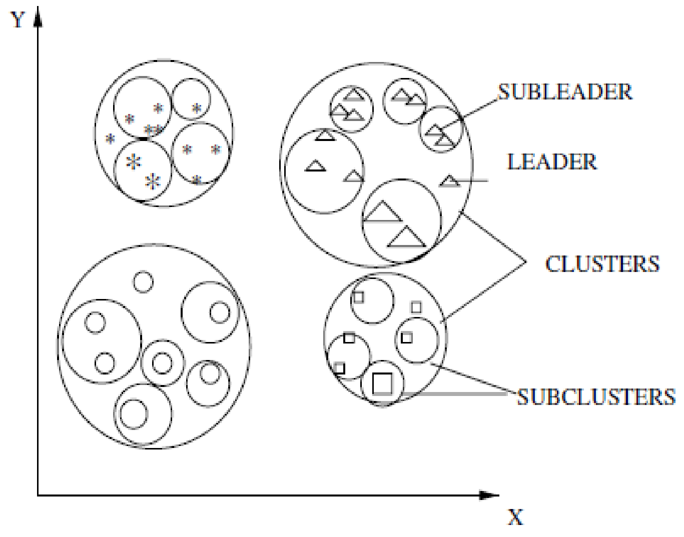


Fig. 10. Clusters in Leaders–Subleaders algorithm [19]

5.7 Bisecting k-means

Bisecting k-means (BKMS) is a divisive clustering algorithm. It is proposed by Steinbach et al. (2000) in the context of document clustering. Bisecting k-means always finds the partition with the highest overall similarity, which is calculated based on the pair wise similarity of all points in a cluster. This procedure will stop until the desired number of clusters is obtained. As reported, the bisecting k-means frequently outperforms the standard k-means and agglomerative clustering approaches. In addition, the bisecting k-means' time complexity is O(nk) where n is the number of items and k is the number of clusters. Advantage of BKMS is low computational cost. BKMS is identified to have better performance than k-means (KMS) agglomerative hierarchical algorithms for clustering large document [7].

6. Comparison of algorithms

In section 5, we introduced some hierarchical clustering algorithms. Now in this section we compare them with each other according to similarity and difference.

BIRCH and CURE, both of which utilize only the cluster centroid for the purpose of labeling. CURE's execution times are always lower than BIRCH's [9]. CURE ignores the information about the aggregate inter-connectivity of objects in two clusters. so it is introduce Chameleon algorithm[5]. Among all of algorithms that we introduce in this paper, BIRCH sensitive to the order of the data record[5]. Both of BIRCH and Leaders–Subleaders are incremental [19,20]. CHAMELEON solve two great weakness of hierarchical clustering algorithms: inter-connectivity of two clusters, which is in CURE algorithm; closeness of two clusters, which is in ROCK algorithm [5]. The algorithms CURE and ROCK are based on "static" modeling of the clusters but CHAMELEON is an efficient algorithm that uses a dynamic model to obtain clusters [2, 3]. The clusters produced by the single link algorithm are formed at low dissimilarities in the dissimilarity dendrogram. On the other hand, the clusters produced by the complete link algorithm are formed at high dissimilarities in the dissimilarity dendrogram [3]. Table 1 shows some features of Hierarchical clustering algorithms that introduced in section 5.

Table 1. Comparison of hierarchical clustering algorithms

Algorithms	*Hierarchical*		*For large data set*	*Sensitive to Outlier/Noise*	*Model*		*Time complexity*	*Space complexity*
	agglomerative	divisive			Static	Dynamic		
CURE	✓		✓	**Less sensitive to noise**	✓		$O(n^2 logn)$	$O(n)$
BIRTH	✓		✓	**Handle noise effectively**		✓	$O(n)$	–
ROCK	✓		✓	-	✓		$O(n2+m_m m_a+n^2 logn)$	$O(min\{n^2,nm_m m_a\})$
Chameleon	✓		✓	-		✓	$O(n(log_2 n +m))$	–
S-link	✓		–	**Sensitive to outlier**	-	–	$O(n^2 logn)$	$O(n^2)$
Ave-link	✓		-	-	–	–	–	–
Com-link	✓			**Not strongly affected by outliers**	–	–	$O(n^3)$ **: obvious algorithm**	–
							$O(n^2 logn)$ **: priority queues**	$O(n^2)$
							$O(n^2)$	$O(n)$
							$O(nlog^2 n)$ **: Euclidean plan**	$O(n)$
							$O(n\ logn + nlog^2(1/\varepsilon)$ **:ε- approximation**	$O(n)$
Leader-Subleaders	✓		✓	-	–	–	$O(ndh)$ **:h=2**	$O((L+SL)d)$
BKMS		✓	–	-	–	–	$O(nk)$	–

d: dimensionality of the data point, h: number of level, k: number of clusters , L: number of Leader,*m:* number of clusters formed after completion of the first phase of the CHAMELEON algorithm, m_a : average number of neighbors for a point, m_m maximum number of neighbors for a point, n:number of data point, SL: number of Subleaders.

7. Conclusion

In this paper, we classified clustering algorithms, and then focused on hierarchical clustering algorithms. One of the most purposes of algorithms to minimize disk I/O operations, consequently reducing time complexity. We have declared algorithms attributes, Disadvantages and advantages. Finally, we compare them.

Acknowledgment. The authors would like to express their thanks to referees for their comments and suggestions which improved the paper.

References.

[1] A.K. Jain, M.N. Murty and P.J. Flynn, Data clustering: A review, ACM Computing Surveys, 31 (1999), 264-323.

[2] N. A. Yousri, M. S. Kamel and M. A. Ismail, A distance-relatedness dynamic model for clustering high dimensional data of arbitrary shapes and densities, Pattern Recognition, 42 (2009), 1193-1209.

[3] K. Koutroumbas and S. Theodoridis, Pattern Recognition, Academic Press, (2009).

[4] M. Kantardzic ,Data Mining: Concepts, Models, Methods, and Algorithms, John Wiley & Sons, (2003).

[5] R. Capaldo and F. Collova, Clustering: A survey, Http://uroutes.blogspot.com, (2008).

[6] D.T. Pham and A.A. Afify, Engineering applications of clustering techniques, Intelligent Production Machines and Systems, (2006), 326-331.

[7] L. Feng, M-H Qiu, Y-X. Wang, Q-L. Xiang, Y-F. Yang and K. Liu, A fast divisive clustering algorithm using an improved discrete particle swarm optimizer, Pattern Recognition Letters, 31 (2010), 1216-1225.

[8] R. Gil-García and A. Pons-Porrata, Dynamic hierarchical algorithms for document clustering, Pattern Recognition Letters, 31 (2010), 469-477.

[9] S. Guha, R. Rastogi and K. Shim, CURE: An efficient clustering algorithm for large databases, Information Systems, 26 (2001), 35-58.

[10] J.A.S. Almeida, L.M.S. Barbosa, A.A.C.C. Pais and S.J. Formosinho, Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering, Chemometrics and Intelligent Laboratory Systems, 87 (2007), 208-217.

[11] M. Charikar, C. Chekuri, T. Feder and R. Motwani, Incremental Clustering and Dynamic Information Retrieval, Proceeding of the ACM Symposium on Theory of Computing, (1997), 626-634.

[12] T. Zhang, R. Ramakrishnan and M. Livny, BIRCH: An efficient clustering method for very large databases, Proceeding of the ACM SIGMOD Workshop on Data Mining and Knowledge Discovery, (1996), 103-114.

[13] J. Harrington and M. Salibián-Barrera, Finding approximate solutions to combinatorial problems with very large data sets using BIRCH, Computational Statistics and Data Analysis, 54 (2010), 655-667.

[14] M. Dutta, A. Kakoti Mahanta and A.K. Pujari, QROCK: A quick version of the ROCK algorithm for clustering of categorical data, Pattern Recognition Letters, 26 (2005), 2364-2373.

[15] S. Guha, R. Rastogi and K. Shim, ROCK: A robust clustering algorithm for categorical attributes, Information Systems, 25 (2000), 345-36.

[16] G. Karypis, E.H. Han and V. Kumar, CHAMELEON: Hierarchical clustering using dynamic modeling, IEEE Computer, 32 (1999), 68-75.

[17] Y. Song, S. Jin and J. Shen, A unique property of single-link distance and its application in data clustering, Data & Knowledge Engineering, 70 (2011), 984-1003.

[18] D. Krznaric and C. Levcopoulos, Optimal algorithms for complete linkage clustering in d dimensions, Theoretical Computer Science, 286 (2002), 139-149.

[19] P.A. Vijaya, M. Narasimha Murty and D.K. Subramanian, Leaders–Subleaders: An efficient hierarchical clustering algorithm for large data sets, Pattern Recognition Letters, 25 (2004), 505-513.

[20] V.S. Ananthanarayana, M. Narasimha Murty and D.K. Subramanian, Rapid and Brief Communication Efficient clustering of large data sets, Pattern Recognition, 34 (2001), 2561-2563.