Analyzing Popular Clustering Algorithms from Different Viewpoints[*]

QIAN Wei-ning, ZHOU Ao-ying

(*Department of Computer Science, Fudan University, Shanghai* 200433, *China*)

(*Laboratory for Intelligent Information Processing, Fudan University, Shanghai* 200433, *China*)

E-mail: {wnqian,ayzhou}@fudan.edu.cn

http://www.cs.fudan.edu.cn/ch/third_web/WebDB/WebDB_English.htm

Received September 3, 2001; accepted February 25, 2002

Abstract: Clustering is widely studied in data mining community. It is used to partition data set into clusters so that intra-cluster data are similar and inter-cluster data are dissimilar. Different clustering methods use different similarity definition and techniques. Several popular clustering algorithms are analyzed from three different viewpoints: (1) clustering criteria, (2) cluster representation, and (3) algorithm framework. Furthermore, some new built algorithms, which mix or generalize some other algorithms, are introduced. Since the analysis is from several viewpoints, it can cover and distinguish most of the existing algorithms. It is the basis of the research of self-tuning algorithm and clustering benchmark.

Key words: data mining; clustering; algorithm

Clustering is an important data-mining technique used to find data segmentation and pattern information. Clustering technique is widely used in applications of financial data classification, spatial data processing, satellite photo analysis, and medical figure auto-detection etc.. The problem of clustering is to partition the data set into segments (called clusters) so that intra-cluster data are similar and inter-cluster data are dissimilar. It can be formalized as follows:

Definition 1. Given a data set $V\{v_1,v_2,...,v_n\}$, in which v_i's (i=1,2,…,n) are called data points. The process of partitioning V into $\{C_1,C_2,...,C_k\}$, $C_i\subseteq V$ (i =1,2,…,k), and $\cup_{i=1}^{k} C_i = V$, based on the similarity between data points are called clustering, C_i's (i =1,2,…,k) are called clusters.

The definition does not define the similarity between data points. In fact, different methods use different criteria.

Clustering is also known as unsupervised learning process, since there is no priori knowledge about the data set. Therefore, clustering analysis usually acts as the preprocessing of other KDD operations. The quality of the clustering result is important for the whole KDD process. As other data mining operations, high performance and scalability are other two requests beside the accuracy. Thus, a good clustering algorithm should satisfy the following

[*] Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1998030414 (国家重点基础研究发展规划973项目); the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.99038 (国家教育部博士点基金)

QIAN Wei-ning was born in 1976. He is a Ph.D. candidate at the Department of Computer Science, Fudan University. His research interests are clustering, data mining and Web data management. **ZHOU Ao-ying** was born in 1965. He is a professor and doctoral supervisor at the Department of Computer Science, Fudan University. His current research interests include Web data management, data mining, and object management over peer-to-peer networks.

requests: Independent of in-advance knowledge; Only need easy-to-set parameters; Accurate; Fast; Having good scalability.

Much research work has been done on building clustering algorithms. Each uses novel techniques to improve the ability of handling certain characteristic data sets. However, different algorithms use different criteria as mentioned above. Since there is no benchmark for clustering methods, it is difficult to compare these algorithms by using a common measurement. However, a detailed comparison is necessary. This is because that: (1) The advantages and disadvantages should be analyzed, so that improvement can be developed on existing algorithms. (2) The user should be able to choose right algorithm for a certain data set, so that the optimal result and performance can be obtained. (3) The detailed comparison is the basis for building a clustering benchmark.

In this paper, we analyze several existing popular algorithms from some different aspects. It is different with some other survey work[1~3] in that we compare these algorithms universally from different viewpoints, while others try to generalize some methods to a certain framework, such as in Refs.[1,2], which can only cover limited algorithms, or just introduce clustering algorithms one by one as tutorial[3], so that no comparison among algorithms is analyzed. Since different algorithms use different criteria and techniques, those surveys can only cover some of the algorithms. Furthermore, some algorithms cannot be distinguished since they use a same technique so that they fall into the same category in a certain framework.

The rest of this paper is organized as follows: Section 1 to 3 analyze the clustering algorithms from three different viewpoints, namely, clustering criteria, algorithm framework and cluster representation. Section 4 introduces some methods, which are mixture or generalization of other algorithms. Section 5 introduces research focus on auto-detection of clusters. Finally, Section 6 is for conclusion remarks.

It should be note that from each viewpoint, although we try to classify as many algorithms as we can, someone is still missing. And some algorithms may fall into the same category. However, while we observing these algorithms from all these viewpoints, different algorithms can be distinguished. This is the motivation of our work.

1 Criteria

The basis of clustering analysis is the definition of similarity. Usually, the definition of similarity contains two parts: (1) The similarity between data points; (2) The similarity between sets of data points. Not all clustering methods need both of them. Some algorithms only use one.

The clustering criteria can be classified into three categories: distance-based, density-based, and linkage-based. Distance-based and density-based clustering is usually applied to data in Euclidean space, while linkage-based clustering can be applied to data in arbitrary metric space.

1.1 Distance-Based clustering

The basic idea of distance-based clustering is that a cluster is the data points close to each other. The distance between two data points is easy to define in Euclidean space. The widely used distance definitions include Euclidean distance, and Manhattan distance.

However, there are several choices for similarity definition between two sets of data points, as follows:

$$Similarity_{\mathrm{rep}}(C_i,C_j)=distance(rep_i,rep_j) \tag{1}$$

or

$$Similarity_{\mathrm{avg}}(C_i,C_j)=\frac{1}{n_i\times n_j}\sum_{v_i\in C_i,v_j\in C_j}distance(v_i,v_j) \tag{2}$$

or

$$Similarity_{\max}(C_i, C_j) = \max\{distance(v_i, v_j) \mid v_i \in C_i, v_j \in C_j\} \tag{3}$$

or

$$Similarity_{\min}(C_i, C_j) = \min\{distance(v_i, v_j) \mid v_i \in C_i, v_j \in C_j\} \tag{4}$$

In (1), rep_i and rep_j are representatives of C_i and C_j, respectively. The representative of a data set is usually the mean, such as in *k*-means [4]. Single representative methods usually employ Definition (1). It is obvious that the complexity of (2), (3), and (4) are all $O(|C_i|*|C_j|)$, which are inefficient for large data sets. Although they are more global definitions, they are usually not directly applied on similarity definition for sub-clusters or clusters. The only exception is BIRCH[5], in which CF-vector and CF-tree are employed to accelerate the computation. Some trade-off approaches are taken, as it will be discussed in Section 2.1, in which the detailed analysis of single representative methods is also given.

The advantage of distance-based clustering is that distance is easy for computing and understanding. And distance-based clustering algorithms usually need parameters of *K*, which is the number of final clusters user wants, or the minimum distance to distinguish two clusters. However, the disadvantage of them is also distinct that they are noise-sensitive. Although some techniques are introduced in some of them, they result in other serious problems. CURE[6] uses representative-shrinking techniques to reduce the impact of noises. However, it invites the problem that it fails to identify the clusters in hollow shapes, as the result in our experiment shown in Fig.1. This shortcoming counteracts the advantage of multi-representatives that the algorithm can identify arbitrary-shaped clusters. BIRCH, which is the first clustering algorithm considering noises, introduces a new parameter *T*, which is substantially a parameter related to density. Furthermore, it is hard for user to understand this parameter unless the page storage ability of CF-tree is known(Page_size/entry_size/T is an approximation of density in that page). In addition, it may cause loss of small clusters and long-shaped clusters. Since lack of space, the detailed discussion is omitted here.

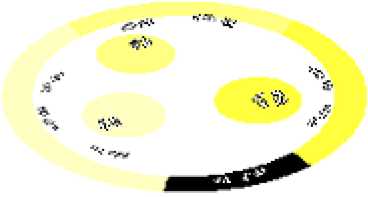

Fig. 1 Hollow-Shaped cluster identified by CURE

1.2 Density-Based clustering

Other than distance-based clustering methods, density-based clustering stands for that clusters are dense areas. Therefore, the similarity definition of data points is based on whether they belong to connected dense regions. The data points belonging to the connected dense region belong to the same cluster. Based on the different computation of density, density-based clustering can be further classified into Nearest-Neighbor (called NN in the rest of this paper) methods and cell-based methods. The difference between them is that the former define density based on data set, and the latter define it based on data space. No matter which kind a density-based clustering algorithm belongs to, it always needs a parameter of minimum-density threshold, which is the key to define dense region.

1.2.1 NN methods

NN methods only treat points, which have more than *k* neighbors in hyper-sphere whose radius is *ε*, as data points in clusters. Since the neighbors of each point should be counted, the index structures which support region query, such as R*-tree, or X-tree, are always employed. Because of the curse of dimensionality[7], these methods don’t have good scalability for dimensionality. Furthermore, NN methods will result in frequent I/O when the data sets are very large. However, for most multi-dimensional data sets, these methods are efficient. In short, the shortcoming of this kind of methods is the shortcoming of the index structures they based-on.

Traditional NN methods, such as DBSCAN and its descendants[8~10], need parameters of density threshold and ε. Recently, OPTICS[11], whose basic idea is the same as DBSCAN, focuses on automatically identification of cluster structures. Since the novel techniques in OPTICS do not belong to the topic of this sub-section, we will discuss them in Section 5.

1.2.2 Cell-Based methods

Cell-based methods count density information based on the units. STING[12], WaveCluster[13], DBCLASD[14], CLIQUE[15], and OptiGrid[16] all fall into this category. Cell-based methods have the shortcoming that cells are only pproximation of dense areas. Some methods introduce techniques to solve this problem, as will be introduced in Section 2.3.

Density-based clustering methods all meet problem when data sets contain clusters or sub-clusters whose granularity is smaller than the granularity of units for computing density. A well-known example is the dumbbell-shaped clusters, as shown in our experimental result, Figure 2. However, for density-based clustering methods, it is easy to remove noises, if the parameters are properly set. That is to say, it is robust to noises.

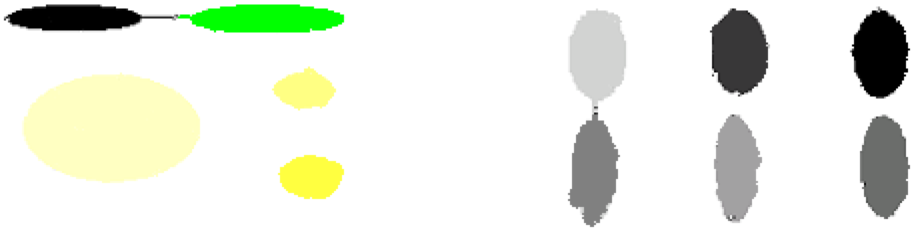

Fig.2 Dumbbell-Shaped clusters identified by density-based algorithm (DBSCAN)

1.3 Linkage-Based clustering

Other than distance-based or density-based clustering, linkage-based clustering can be applied to arbitrary metric spaces. Furthermore, since in high-dimensional space, the distance information and density information is not sufficient for clustering, linkage-based clustering is often employed. Algorithms belonging to this kind include ROCK[17], CHAMELEON[18], ARHP[19,20], STIRR[21], CACTUS[22], etc.

Linkage-based methods are based on graph or hyper-graph model. They usually map the data set into a graph/hyper-graph, then cluster the data points based on the edge/hyper-edge information, so that the highly connected data points are assigned to the same cluster. The difference between graph model and hyper-graph model is that the former reflects the similarity of pair of nodes, while the latter usually reflects the co-occurrence information. ROCK and CHAMELEON use graph model, while ARHP, PDDP, STIRR, and CACTUS use hyper-graph model. Although the developers of CACTUS didn’t state that it is a hyper-graph-model-based algorithm, it belongs to that kind.

The quality of linkage-based clustering result depends on the definition of link or hyper-edge. Since it is impossible to handle a complete graph, the graph/hyper-graph model always eliminates the edges/hyper-edges whose weight is low, so that the graph/hyper-graph is sparse. However, to gain the efficiency, it may reduce the accuracy.

The algorithms fall in this category use different frameworks. ROCK and CHAMELEON are hierarchical clustering methods, while ARHP is divisive method, and STIRR uses dynamical system model. Furthermore, since the co-occurrence problem is similar to association rule mining problem, ARHP and CACTUS both borrow Apriori algorithm[23] to find the clusters. Another algorithm employ Apriori-like algorithm is CLIQUE. However, the monotonicity lemma is used to find high-dimensional clusters based on clusters find in subspaces. CLIQUE is not linkage-based clustering methods, which is the difference between it with other algorithms discussed in this subsection. The detailed discussion of algorithm framework will be given in Section 3. And since CHAMELEON uses both link and distance information, it will be discussed standalone in Section 4.1.

2 Cluster Representation

The purpose of clustering is to identify the data clusters, which are the summary of the similar data. Each algorithm should represent the clusters and sub-clusters in some forms. Although labeling each data point with a cluster identity is a straightforward idea, most methods don’t employ this approach. This may be because that: (1) The summary, which should be easily understandable, is more than (data-point, cluster-id) pairs; (2) It is time- and space-expensive to label all the data points in the process of clustering; (3) Some methods employ accurate compact cluster representatives, which make the time-consuming process of labeling unnecessary. We classify the cluster representation techniques into four kinds, as discussed in the following:

2.1 Representative points

Most distance-based clustering methods use some points to represent clusters. These points are called representative points. The representatives may be data points, or some other points that do not exist in database, such as means of some sets of data points. The data representation techniques falling into this category can be further classified into three classes:

2.1.1 Single representative

The simplest approach is to use one point as the representative of each cluster. Each data point is assigned to the cluster whose representative is the closest one. The representative point may be the mean of the cluster, like *k*-means[4] methods do, or the data point in the database, which is the closest point to the center, like *k*-medoids methods do. Other algorithms fall into this kind include BIRCH[5], CLARA[24], and CLARANS[25]. The different affect of *k*-means and *k*-medoids methods on clustering result is introduced in detail in Ref.[25]. Since it is not related to the motivation of this paper, we don’t survey it here.

The shortcoming of single representative approach is obvious: (1) only sphere clusters can be identified; and (2) large clusters with small cluster beside will be split, while some data points in the large cluster will be assigned to the small cluster. These two conditions are shown in Fig.3 (The right part of this Figure is borrowed from Ref.[6], Fig.1(b)). Therefore, this approach will fail when processing data sets with arbitrary shaped clusters or clusters with great difference.

2.1.2 All data points

Using all the data points in a cluster to represent it is another straightforward approach. However, it is time-expensive since: (1) the data sets are always large so that the label information cannot fit in memory, which leads to frequent disk access, and (2) while computing information intra- and inter- clusters, it will access all data points. Furthermore, the label information is hard to understand. Therefore, no popular algorithms take this approach.

2.1.3 Multi-Representatives

Multi-representatives approach is introduced in CURE, which is the trade-off between single-point and all-points methods. The first representative is the data point, which is the farthest to the mean of the cluster. And next, the data point, whose distance to the nearest existing representative is the largest, is chosen each time, until the number of representatives is large enough. In Ref.[6], the experiments show that for most data sets, 10

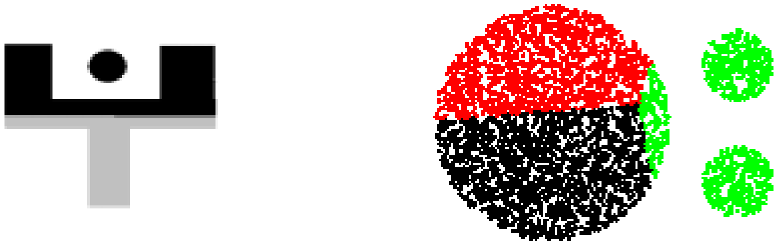

Fig.3 Non-Spherical clusters and clusters with different scales identified by single representative methods

representatives will lead to satisfied result. In the long version of Ref.[26], the authors who developed CURE also mentioned that for complex data sets, more representatives are needed.

However, before clustering, the complexity of the clusters is unknown. Furthermore, the relationship between complexity of clusters and number of representatives is not clear. This forces the user to choose a large number of representatives. Since the time complexity of CURE is $O(n^2\log n)$, in which n is the number of data points in the beginning, the existence of large number of representatives in the initial sub-clusters will affect the efficiency (there exists sub-clusters because that a simple partitioning technique is used in CURE[6]. The time-complexity according to number of representatives is $O(c{*}\log c)$, if the number of initial sub-clusters is a fixed number), as shown in our experimental result, Fig.4. Furthermore, along with the technique they handling outliers (the shrinking of representatives), it fails to identify clusters of hollow shape, as it has already been discussed in Section 1.1 and shown in Fig.1. However, it outperforms single-point and all-points approaches when both effectiveness and efficiency are considered.

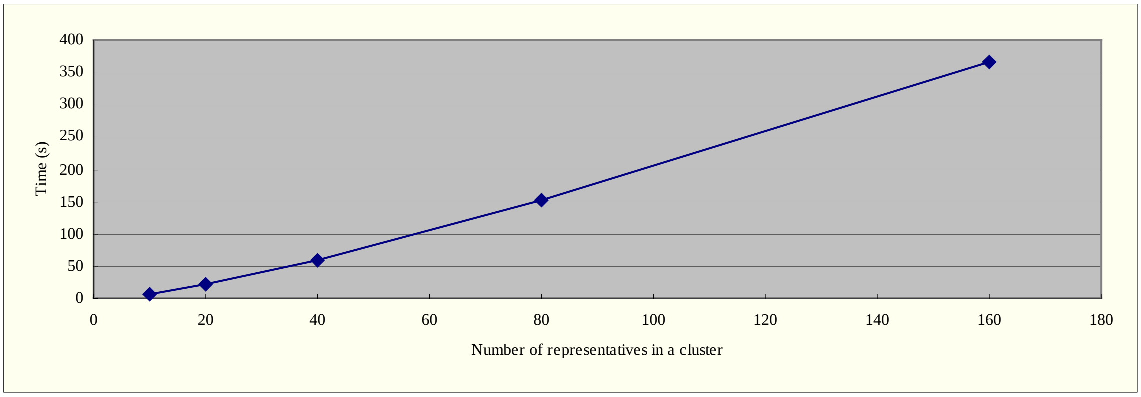

Fig. 4 Performance of CURE vs. number of representatives in a cluster

2.2 Dense area

Some density-based clustering algorithms use dense area to denote clusters and sub-clusters. DBSCAN[8], its descendants[9,10], and OPTICS[11] belong to this category. Dense area representation method is similar to all-data-points methods except that only core points are used. Core points are those data points whose neighbors within a certain region are more than the threshold. Therefore, only core points are used to expand a sub-cluster, and it will stop when no further expansion can be applied on core points.

Dense area can figure arbitrary-shaped clusters besides the dumbbell-shaped clusters. However, the cost for computing core points is expensive, so that special index structures are needed. In algorithms of DBSCAN series and OPTICS, R^*-tree is used to support region query. Since these methods need to scan the whole database, and each point may cause a region query, these methods always result in frequent I/O when applied to large databases, as shown in experiments given in Section 4.2.

2.3 Cells

Some grid-based methods use cells to summary the clusters, such as STING[12], WaveCluster[13], CLIQUE[15], DBCLASD[14], and OptiGrid[16] etc..

Other than dense areas, which are the condensation of dense data points, cells are partitions of the data space. Therefore, a cell is the approximation of the data points falling into it. This makes the algorithms taking this approach inaccurate in some condition. In Ref.[12], the authors argue that under a sufficient condition, STING can ensure the result is accurate. However, this conclusion is made in the condition that the characteristic of queries is known a priori. WaveCluster facilitates the multi-resolution property of wavelet to identify clusters in different resolutions, which ensure that the highest resolution clusters are accurate.

The advantage of using cells to represent clusters is straightforward. Firstly, the number of cells is much smaller than the size of the database. Therefore, the data for processing is limited, which leads to high scalability of those approaches. Secondly, the cost of computing properties of cells is low compared to finding dense area, which needs complex data structure support. This is because that cells are data independent, while dense area depends on data distribution. At last, as dense areas, cells can reflect the data distribution information of a local area, although it is approximate.

Since the number of neighboring relationship is explosive when the dimensionality is increasing, the algorithms facilitating neighboring information of cells is usually inefficient for high-dimensional data. The only exception is CLIQUE. Different from other cell representation methods, CLIQUE finds dense units (cells) from low-dimensional subspaces to high-dimensional subspaces. Therefore, it has high scalability to dimensionality.

Although OptiGrid is a cell-based clustering method, it does not use neighboring information, so that it is efficient for high-dimensional data sets.

2.4 Probability

Some methods use probability to denote the degree of a data points belonging to a cluster. EM[27,28], and AutoClass[29] belong to this category. The problem of classifying a data point to more than one cluster is also known as fuzzy clustering or soft clustering. In most cases, the performance of soft clustering is unsatisfactory. Reference [2] provides a detailed survey of fuzzy clustering. Since the lack of space, we are not verbose here.

3 Algorithm Framework

In the above two sections, we discussed the clustering criteria and cluster representation, which are the two most important factors for clustering effectiveness. In this section, the algorithm framework will be discussed. The algorithm framework determines the time complexity of the algorithms, and the needed parameters. Furthermore, algorithm framework also affects the techniques of preprocessing. These are the focuses in the following three subsections.

3.1 Optimization methods

Optimization methods usually try to optimize a certain measure. Traditional optimization methods are also known as partitioning methods. The most famous ones include *k*-means (including its variance *k*-modes[30], *k*-prototypes[30])[4], and *k*-medoids (including PAM[24], CLARA[24], CLARANS[25], etc.). Some new built algorithms also fall into this category, including STIRR[21].

K-means methods try to minimize a dissimilar criterion (typically the squared-error criterion). *K*-means algorithms usually are linear to the size of the data set. However, they are usually sensitive to outliers, and often terminate at a local optimum. Therefore, the quality of the result is not satisfiable. Furthermore, they are usually designed as memory-resident algorithms, which limits the scalability.

Other than *k*-means, *k*-medoids methods use data points to represent a cluster. Since noises or outliers less influence the medoids, they are more robust than *k*-means. However, the cost of *k*-medoids algorithms is also expensive. PAM, CLARA, and CLARANS are three most famous *k*-medoids algorithms. PAM is the first *k*-medoids method. CLARA and CLARANS both use sampling technique, in which CLARA use fixed samples, while CLARANS don't. Furthermore, CLARANS exploits randomized search. Therefore, CLARANS is more scalable than PAM and CLARA.

Other than *k*-means or *k*-medoids, some new built optimization algorithms don't use representatives, such as STIRR. STIRR is designed to handle categorical data, so that means or medoids is difficult to define. It maps the data set into a hyper-graph and then employs dynamical system techniques to find basins, which are fix-points of the system. Therefore, it can be viewed as the process of finding an optimum of the system configuration.

3.2 Agglomerate methods

Agglomerate algorithms treat data points or data set partitions as sub-clusters in the beginning. Then they merge the sub-clusters iteratively until the final clusters are gotten. BIRCH[5], CURE[6], ISAAC[31], ROCK[17], STING[12], CHAMELEON[18], all fall into this category.

The agglomerate methods have the shortcoming that the time complexity is at least $O(n2)$. Therefore, several techniques are employed to accelerate the processing. Since the number of the merge operations depends on the number of initial objects, some preprocessing techniques are used to reduce the object to be processed. Sampling and partitioning are two widely used preprocessing techniques. The developers of CURE proved that a small sample could guarantee the quality of clustering, while CURE, STING, CHAMELEON all use partitioning before merging the sub-clusters. Another technique used to accelerate the processing is indexing. Nearly all agglomerate algorithms exploit special index structure. BIRCH uses CF-tree, CURE uses *k*-d-tree and heap, ROCK uses two-level heap, STING uses quad-tree-like index, and CHAMELEON uses *k*-d-tree and heap-based priority queue.

Agglomerate methods usually need a parameter known as stop condition, which is used to determine when the merge operations should stop. This parameter may be *k*, the number of final clusters, or a threshold, which denotes the minimum value of the merging measurement.

3.3 Divisive methods

Divisive methods belong to hierarchical methods as agglomerate methods do. Divisive methods begin with a large cluster, which contains all the data points, and then partition the cluster based on the dissimilarity recursively, until some stop condition is reached. ARHP[19,20], PDDP[20], and OptiGrid[16] fall into this category.

ARHP uses hyper-graph model. The whole data set is mapped to a hyper-graph by using association rule discovery techniques first. Then, the sub-graphs satisfy that the fitness is larger than a threshold is partitioned out. At last, the vertices are assigned to the clusters they are highly connected to. Other than ARHP, which uses fitness to partition the clusters, PDDP and OptiGrid use a hyper-plane to split a cluster in each iteration.

As agglomerate methods, divisive methods also need the parameter of stop condition. It can be either the number of final clusters: *k*, or a threshold for partitioning, such as fitness-threshold. The advantage of divisive methods is that, for graph/hyper-graph model, there is some mature research work, such as HMETIS[32], can be employed. In fact, even CHAMELEON[18], an agglomerate method, has a divisive step as the pre-processing to get the initial sub-clusters. Since it is the preprocessing, the parameter is easy to set.

4 Mixed or Generalized Clustering Approaches

As analyzed above, algorithms using single criteria may fall down on handling some kind of data sets. Some recent research focuses on combining or generalizing different criteria. In this section, three algorithms of this kind will be introduced and analyzed.

4.1 CHAMELEON: distance + connectivity method

CHAMELEON[18] is an algorithm combining several existing clustering techniques. From the clustering criteria viewpoint, it combines distance measurement (relative closeness) with linkage measurement (relative inter-connectivity). Furthermore, it generalizes the classic distance measurement in that it uses relative criteria, which is first introduced in linkage-based clustering[19]. From the algorithm framework viewpoint, it uses divisive method as partitioning step to generate the initial sub-clusters. And the main phase of the algorithm employs agglomerate framework. From the cluster representation viewpoint, it is an all-point method. However, the ‘points’ here may be the initial sub-clusters.

The advantages and shortcomings of CHAMELEON can be derived easily from the multiple viewpoints analysis. It is strong at identification of arbitrary shaped clusters and highly intra-connective clusters, since relative distance and relative connectivity are used. However, it needs two parameters as the threshold of relative distance and relative connectivity respectively. Furthermore, the divisive partitioning needs another parameter. This is the shortcoming of combining so many techniques together. Furthermore, the framework determines that index structure (e.g. k-d-tree) supports region query and a heap must be used. Although the time complexity is analyzed theoretically, the scaling up technique or experiment is not provided in the paper.

4.2 Hybrid: distance + density method

Hybrid algorithm is a clustering method combining distance and density criteria[33]. From the viewpoint of criteria, it uses distance and density information. From the cluster representation viewpoint, it uses multi-representative technique. Although cell is employed to enable the scaling up processing, it is not used to present the clusters, so that the cluster representation could be more accurate. From the framework viewpoint, it is an agglomerate algorithm.

As discussed before, the advantages and shortcomings is straightforward after the analysis. It can identify arbitrary-shaped clusters, and be insensitive to noises or outliers, since both distance and density information are taken use of. However, this introduced three parameters: one is for distance computing while other two are for density computing. Furthermore, the framework determines the use of k-d-tree and heap structure. Different from CHAMELEON, it is designed to handling very large databases. The cell-based indexing not only reduces the data to be processed, but also accelerating the labeling process. As shown in our experiments, Fig.5, it outperforms two popular clustering algorithms DBSCAN and CURE, since that R^*-tree takes high overhead when processing large data sets, while CURE fails when data sets scales out of the main memory. Detailed description of the experiments can be found in Ref.[33].

4.3 DENCLUE: generalized density method

DENCLUE[34] is a density-based clustering method, which tries to generalize several other clustering algorithms. It can be viewed as a kind of survey on density-based clustering algorithms, since it can cover almost all density-based algorithms by using different influence function and density function. The developers of DENCLUE also state that it can generalize hierarchical algorithms and partitioning algorithms (named as traditional optimization algorithms in this paper). However, it can only denote the framework of those algorithms. It cannot cover those algorithms using representatives, even different functions or parameters are set.

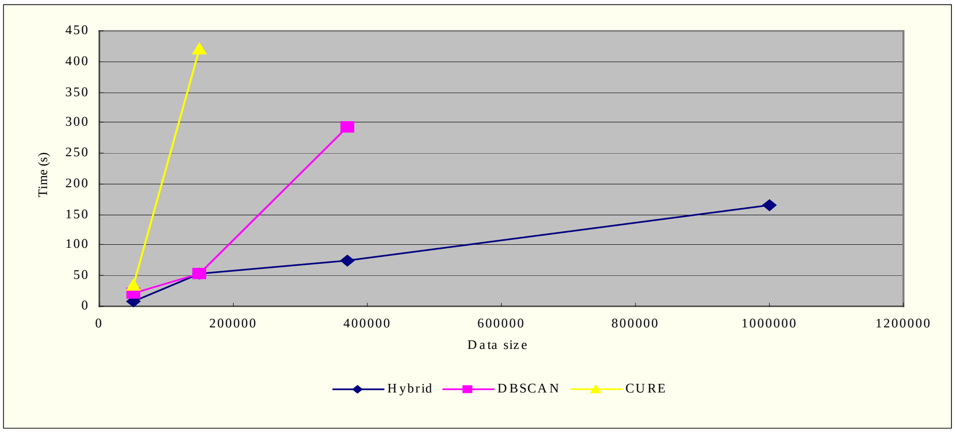

Fig.5 Scaling-up experiments of CURE, DBSCAN, and Hybrid algorithm

Since DENCLUE is in fact a density-based method. It needs to determine the parameters to calculate density, and be robust to noises. Furthermore, the cell-based technique determines that a tree-based index should be taken use of, so that it can handle very large data sets. It also employs a filtering technique to reduce the complexity of handling high-dimensional data. However, another parameter should be introduced.

5 Automatic and Visualization Approaches

Since clustering is a process of unsupervised learning, setting appropriate parameters is a problem for lots of algorithms. The above analysis show that for most clustering algorithms, some parameters are needed. Although they may be straightforward in some cases, they are difficult to set in many environments. Furthermore, current cluster representation techniques can be easily understood only when the data is in low-dimensional space. Therefore, some algorithms are built for automatic clustering. Meanwhile, some other efforts has been made to visualize the process of clustering, so that the user can set the parameters easily and the result can be more understandable.

OPTICS[11] is an algorithm, which is designed to discover cluster structure. It is essentially a density-based clustering algorithm, as DBSCAN is. The difference between OPTICS and other density-based methods is that it uses reachability-plots to visualize the process of clustering. Furthermore, it introduces an automatic technique to detect the steep points, so that clusters can be discovered. By using different parameters, it can discover clusters in different density-level. Therefore, cluster structure is an organization of clusters in different density.

In Ref.[35], the authors introduced an algorithm to build multi-granularity cluster-tree. They argued that an accurate multi-granularity cluster-tree should be vertical distinguished, horizontal distinguished, and complete, which ensure that each node in the cluster-tree denotes a cluster in a certain granularity, while any cluster in any granularity has a corresponding node in the cluster-tree. The construction of multi-granularity cluster-tree employs distance-based clustering in agglomerate framework, which is the main difference between multi-granularity cluster-tree with cluster structure in Ref.[11]. Therefore, clusters in different density will be treated as clusters in different level, and clusters in different scale may be treated as clusters in the same level, by OPTICS; while multi-granularity cluster-tree will treat them in the contrary, as shown in Fig.6. The difference exists because that the motivation of building multi-granularity cluster-tree is to provide a cluster management facility to ease the understanding of clustering result, while OPTICS is designed for automatically or interactive clustering.

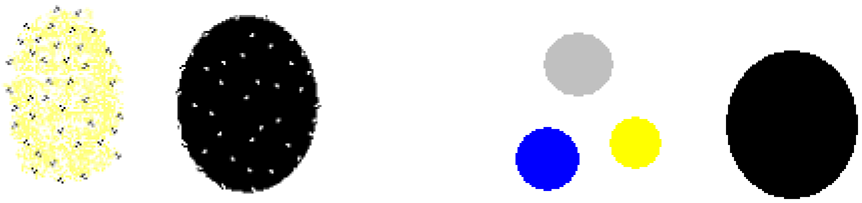

Fig.6

Some researchers in computer graphics also developed some algorithms to visualize the clustering process, such as H-BLOB[36]. However, the basic idea is similar: (1) visualize the clustering processing, so that the construction of clusters can be seen by the user; (2) clusters may exist in different levels, while different parameters are used, whatever which criteria is used.

6 Conclusions

In this paper, we try to analyze the existing popular clustering algorithms both theoretically and experimentally from three different viewpoints: clustering criteria, cluster representation, and algorithm framework, so that most algorithms can be covered, and distinguished. This work can be the basis of: (1) Clustering algorithm advantage/disadvantage analysis; (2) Clustering algorithm selection for data mining users; (3) Clustering algorithm auto-selection for different data sets; (4) Self-tuning clustering algorithm development; (5) Clustering benchmark construction.

The analysis shows that most current algorithms have its shortcomings while being effective or efficient for some special characteristic data sets.

Furthermore, three algorithms, which generalize or mix some other algorithms, are introduced. And they are analyzed from the three viewpoints introduced in this paper. At last, some automatic/visualization algorithms for clustering are introduced. They are the attempts of researchers to push the unsupervised learning process to a more understandable and automatic stage.

Acknowledgement We would like to thank Dr. Wen Jin in Simon Fraser University for his suggestion on the outline and draft of this paper. We also would like to thank Dr. Joerge Sander for providing the source code of DBSCAN, and Ms. Hailei Qian for helping us to implement the algorithms of CURE and Hybrid.

References:

[1] Fasulo, D. An analysis of recent work on clustering algorithms. Technical Report, Department of Computer Science and Engineering, University of Washington, 1999. http://www.cs.washington.edu.

[2] Baraldi, A., Blonda, P. A survey of fuzzy clustering algorithms for pattern recognition. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 1999,29:786~801.

[3] Keim, D.A., Hinneburg, A. Clustering techniques for large data sets – from the past to the future. Tutorial Notes for ACM SIGKDD 1999 International Conference on Knowledge Discovery and Data Mining. San Diego, CA, ACM, 1999. 141~181.

[4] McQueen, J. Some methods for classification and Analysis of Multivariate Observations. In: LeCam, L., Neyman, J., eds. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability. 1967. 281~297.

[5] Zhang, T., Ramakrishnan, R., Livny, M. BIRCH: an efficient data clustering method for very large databases. In: Jagadish, H.V., Mumick, I.S., eds. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data. Quebec: ACM Press, 1996. 103~114.

[6] Guha, S., Rastogi, R., Shim, K. CURE: an efficient clustering algorithm for large databases. In: Haas, L.M., Tiwary, A., eds. Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data. Seattle: ACM Press, 1998. 73~84.

[7] Beyer, K.S., Goldstein, J., Ramakrishnan, R., *et al*. When is 'nearest neighbor' meaningful? In: Beeri, C., Buneman, P., eds. Proceedings of the 7th International Conference on Data Theory, ICDT'99. LNCS1540, Jerusalem, Israel: Springer, 1999. 217~235.

[8] Ester, M., Kriegel, H.-P., Sander, J., *et al*. A density-based algorithm for discovering clusters in large spatial databases with noises. In: Simoudis, E., Han, J., Fayyad, U.M., eds. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96). AAAI Press, 1996. 226~231.

[9] Ester, M., Kriegel, H.-P., Sander, J., *et al*. Incremental clustering for mining in a data warehousing environment. In: Gupta, A., Shmueli, O., Widom, J., eds. Proceedings of the 24th International Conference on Very Large Data Bases. New York: Morgan Kaufmann, 1998. 323~333.

[10] Sander, J., Ester, M., Kriegel, H.-P., *et al*. Density-Based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery, 1998,2(2):169~194.

[11] Ankerst, M., Breunig, M.M., Kriegel, H.-P., *et al*. OPTICS: ordering points to identify the clustering structure. In: Delis, A., Faloutsos, C., Ghandeharizadeh, S., eds. Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. Philadelphia: ACM Press, 1999. 49~60.

[12] Wang, W., Yang, J, Muntz, R. STING: a statistical information grid approach to spatial data mining. In: Jarke, M., Carey, M.J., Dittrich, K.R., *et al*., eds. Proceedings of the 23rd International Conference on Very Large Data Bases. Athens: Morgan Kaufmann, 1997. 186~195.

[13] Sheikholeslami, G., Chatterjee, S., Zhang, A. WaveCluster: a multi-resolution clustering approach for very large spatial databases. In: Gupta, A., Shmueli, O., Widom, J., eds. Proceedings of the 24th International Conference on Very Large Data Bases. New York: Morgan Kaufmann, 1998. 428~438.

[14] Xu, X., Ester, M., Kriegel, H.-P., *et al*. A distribution-based clustering algorithm for mining in large spatial databases. In: Proceedings of the 14th International Conference on Data Engineering. Orlando: IEEE Computer Society Press, 1998. 324~331.

[15] Agrawal, R., Gehrke, J., Gunopulos, D., *et al*. Automatic subspace clustering of high dimensional data for data mining applications. In: Haas, L.M., Tiwary, A., eds. Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data. Seattle: ACM Press, 1998. 94~105.

[16] Hinnebrug, A., Keim, D.A. Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. In: Atkinson, M.P., Orlowska, M.E., Valduriez, P., *et al*., eds. Proceedings of the 25th International Conference on Very Large Data Bases. Edinburgh: Morgan Kaufmann, 1999. 506~517.

[17] Guha, S., Rastogi, R., Shim, K. ROCK: a robust clustering algorithm for categorical attributes. In: Proceedings of the 15th International Conference on Data Engineering. Sydney: IEEE Computer Society Press, 1999. 512~521.

[18] Karypis, G., Han, E.H., Kumar, V. CHAMELEON: a hierarchical clustering algorithm using dynamic modeling. IEEE Computer, 1999,32(8):68~75.

[19] Han, E.H., Karypis, G., Kumar, V., *et al*. Hypergraph based clustering in high-dimensional data sets: a summary of results. Data Engineering Bulletin, 1998,21(1):15~22.

[20] Boley, D., Gini, M., Gross, R., *et al.* Partitioning-Based clustering for web document categorization. Decision Support System Journal, 1999,27(3):329~341.

[21] Gibson, D., Kleinberg, J.M., Raghavan, P. Clustering categorical data: an approach based on dynamical systems. In: Gupta, A., Shmueli, O., Widom, J., eds. Proceedings of the 24th International Conference on Very Large Data Bases. New York: Morgan Kaufmann, 1998. 311~322.

[22] Ganti, V., Gehrke, J., Ramakrishnan, R. CACTUS, clustering categorical data using summaries. In: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining. San Diego: ACM Press, 1999. 73~83.

[23] Agrawal, R., Srikant, R. Fast algorithms for mining association rules. In: Bocca, J.B., Jarke, M., Zaniolo, C., eds. Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94). Santiago: Morgan Kaufmann, 1994. 487~499.

[24] Kaufman, L., Rousseeuw, P.J. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.

[25] Ng, R.T., Han, J. Efficient and effective clustering methods for spatial data mining. In: Bocca, J.B., Jarke, M., Zaniolo, C., eds. Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94). Santiago: Morgan Kaufmann, 1994. 144~155.

[26] Guha, S., Rastogi, R., Shim, K. CURE: an efficient clustering algorithm for large databases. Information System Journal, 1998, 26(1):35~58.

[27] Dempster, A.P., Laird, N.M., Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society(Series B), 1977,29(1):1~38.

[28] Lauritzen, S.L. The EM algorithm for graphical association models with missing data. Computational Statistics and Data Analysis, 1995,19:191~201.

[29] Cheeseman, P., Stutz, J. Bayesian classification (AutoClass): theory and results. In: Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., *et al*., eds. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996. 153~180.

[30] Huang, Z. Extensions to the *K*-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 1998,2:283~304.

[31] Talavera, L., Bejar, J. Efficient construction of comprehensible hierarchical clustering. In: Zytkow, J.M., Quafalou, M., eds. Principles of Data Mining and Knowledge Discovery, Proceedings of the 2nd European Symposium, PKDD’98. LNCS1510, Nantes: Springer-Verlag, 1998. 93~101.

[32] Karypis, G., Aggarwal, R., Kumar, V., *et al*. Multilevel hypergraph partitioning: application in VLSI domain. In: Proceedings of the 34th Conference on Design Automation. Anaheim, CA: ACM Press, 1997. 526~529.

[33] Zhou, A., Qian, W., Qian, H., *et al*. A hybrid approach to clustering in very large databases. In: Cheung, D., Williams, G.J., Li, Q., eds. Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining. LNCS2035, Hong Kong: Springer-Verlag, 2001. 519~524.

[34] Hinneburg, A., Keim, D.A. An efficient approach to clustering in large multimedia databases with noise. In: Agrawal, R., Stolorz, P.E., Piatetsky-Shapiro, G., eds. Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD’98). New York: AAAI Press, 1998. 58~65.

[35] Zhou, A., Qian, W., Qian, H., *et al*. SACT: automatic cluster-tree construction for very large spatial databases. Technical Report, Computer Science Department, Fudan University, 2001. http://www.cs.fudan.edu.cn/ch/third_web/WebDB/wnqian_English.htm.

[36] Sprenger, T.C., Brunella, R., Gross, M.H. H-BLOB: a hierarchical visual clustering method using implicit surfaces. Technical Report No.341, Computer Science Department, ETH Zürich, 2000. ftp://ftp.inf.ethz.ch/pub/publications/tech-reports/3xx/341.pdf.

从多角度分析现有聚类算法

钱卫宁, 周傲英

(复旦大学 计算机科学系,上海 200433)

(复旦大学 智能信息处理开放实验室,上海 200433)

摘要: 聚类是数据挖掘中研究的重要问题之一.聚类分析就是把数据集分成簇,以使得簇内数据尽量相似,簇间数据尽量不同.不同的聚类方法采用不同的相似测度和技术.从以下 3 个角度分析现有流行聚类算法: (1)聚类尺度; (2)算法框架; (3)簇的表示.在此基础上,分析了一些综合或概括了一些其他方法的算法.由于分析从 3 个角度进行,所提出的方法能够涵盖,并区分绝大多数现有聚类算法.所做的工作是自调节聚类方法以及聚类基准测试研究的基础.

关键词: 数据挖掘;聚类分析;算法

中图法分类号: TP311 文献标识码: A