IOSR Journal of Business and Management (IOSR-JBM)
e-ISSN: 2278-487X, p-ISSN: 2319-7668
PP 34-42
www.iosrjournals.org

A Survey On Data Clustering Approaches

Saikat Samanta[1], Siddhartha Chatterjee[2]
[1]*(Department of Computer Science & Engineering, Bengal College of Engineering, India)*
[2]*(Department of Computer Application, DSMS Group of Institutions, India)*

Abstract: *The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. Clustering is the unsupervised classification of patterns into groups.*
However, clustering differences in assumptions and contexts in different communities has made the transfer of useful generic concepts and methodologies slow to occur. This paper presents an overview of pattern clustering methods from a statistical pattern recognition perspective, with a goal of providing useful advice and references to fundamental concepts accessible to the broad community of clustering practitioners.
We present a taxonomy of clustering techniques, and identify cross-cutting themes and recent advances. We also describe some important applications of clustering algorithms .

Keywords : *Clustering , Pattern recognition, Pattern proximity , fuzzy Clustering, Evolutionary Algorithm , incremental clustering.*

I. Introduction

The goal of this survey is to provide a comprehensive review of different clustering techniques in data clustering. Data analysis procedures can be dichotomized as either exploratory or confirmatory, based on the availability of appropriate models for the data source, but a key element in both types of procedures is the grouping, or classification of measurements based on either goodness-of-fit to a postulated model, or natural groupings revealed through analysis. Cluster analysis is the organization of a collection of patterns into clusters based on similarity. Intuitively, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster. An example of clustering is depicted in Figure 1.

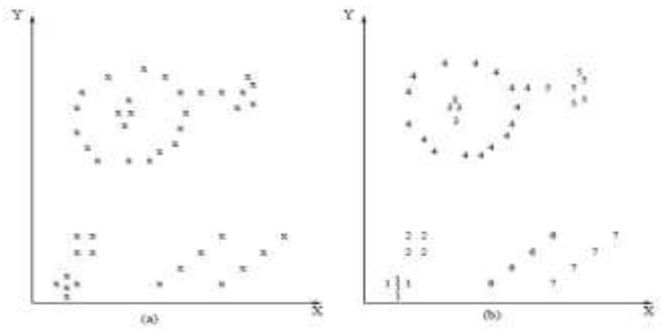

Figure1. Data clustering.

Here, points belonging to the same cluster are given the same label. The variety of techniques for representing data, measuring proximity between data elements, and grouping data elements has produced a rich and often confusing assortment of clustering methods. It is important to understand the difference between clustering and discriminate analysis. Clustering is useful in several exploratory pattern-analysis, grouping, decision- making, and machine-learning situations, including data mining, document retrieval, image segmentation, and pattern classification. The term "clustering" is used in several research communities to describe methods for grouping of unlabeled data. These communities have different terminologies and assumptions for the components of the clustering process and the contexts in which clustering is used. Thus, we face a dilemma regarding the scope of this survey. The production of a truly comprehensive survey would be a monumental task given the sheer mass of literature in this area. The accessibility of the survey might also be questionable given the need to reconcile very different vocabularies and assumptions regarding clustering in the various communities.

Pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in the various. A simple distance measure like Euclidean distance can often be used to reflect dissimilarity between two patterns, whereas other similarity measures can be used to characterize the conceptual similarity between patterns.

The grouping step can be performed in a number of ways. The output clustering can be hard or fuzzy. Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity.

Data abstraction is the process of extracting a simple and compact representation of a data set. Here, simplicity is either from the perspective of automatic analysis or it is human-oriented. In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster prototypes or representative patterns such as the centroid.

II. Clustering Techniques

Different approaches to clustering data can be described with the help of the hierarchy shown in Figure 2

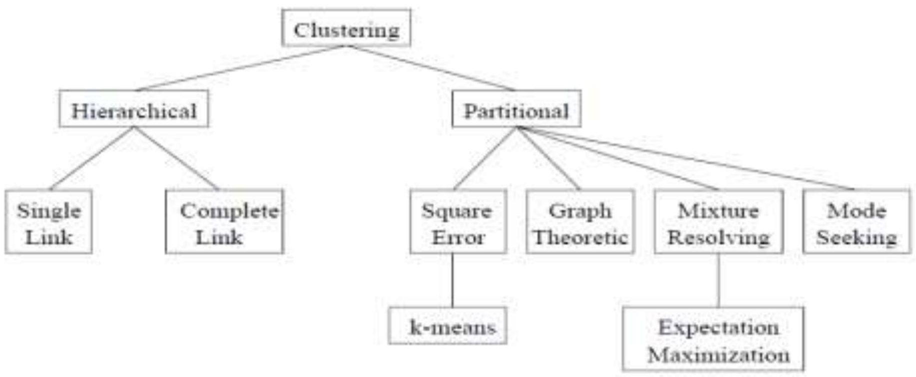

Figure 2. A taxonomy of clustering approaches.

2.1 Hierarchical Clustering Algorithms

The operation of a hierarchical clustering algorithm is illustrated using the two-dimensional data set in Figure 3. This figure depicts seven patterns labeled A, B, C, D, E, F, and G in three clusters.

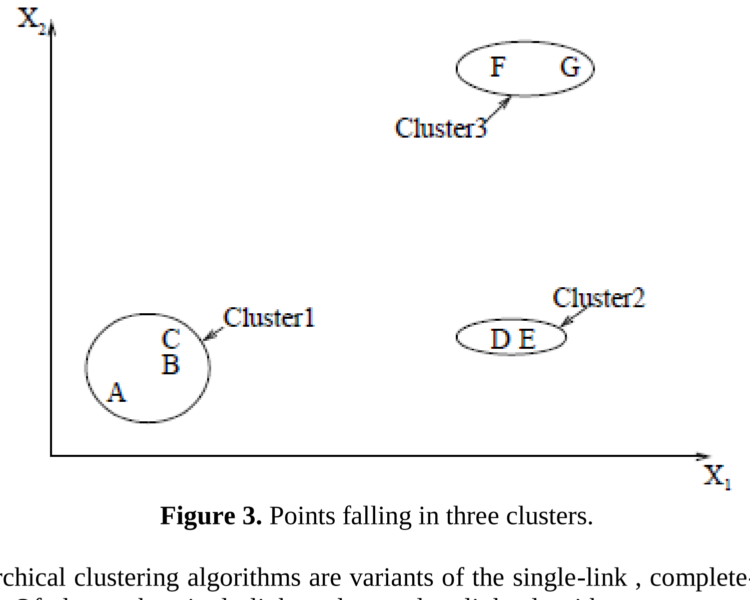

Figure 3. Points falling in three clusters.

Most hierarchical clustering algorithms are variants of the single-link , complete-link , and minimum-variance algorithms. Of these, the single-link and complete-link algorithms are most popular. These two algorithms differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters. In either case, two clusters are merged to form a larger cluster based on minimum distance criteria. The complete-link algorithm produces tightly bound or com-pact clusters.

2.2 Partitional Algorithms

A partitional clustering algorithm obtains a single partition of the data in-stead of a clustering structure, such as the dendrogram produced by a hierarchical technique. Partitional methods have advantages in applications involving large data sets for which the construction of a dendrogram is computationally prohibitive. A problem accompanying the use of a partitional algorithm is the choice of the number of desired output clusters.

2.3 Fuzzy Clustering

Traditional clustering approaches generate partitions; in a partition, each pattern belongs to one and only one cluster. Hence, the clusters in a hard clustering are disjoint. Fuzzy clustering extends this notion to associate each pattern with every cluster using a membership function .The out-put of such algorithms is a clustering, but not a partition. We give a high-level partitional fuzzy clustering algorithm below.

Fuzzy Clustering Algorithm -

(1) Select an initial fuzzy partition of the N objects into K clusters by selecting the $N \times K$ membership matrix U. An element Uij of this matrix represents the grade of membership of object xi in cluster cj. Typically, $u_{ij} \in [0,1]$.

(2) Using U, find the value of a fuzzy criterion function, e.g., a weighted squared error criterion function, associated with the corresponding partition. One possible fuzzy criterion function is

$$E^2(\mathscr{X}, \mathbf{U}) = \sum_{i=1}^{N}\sum_{k=1}^{K} u_{ij}\|\mathbf{x}_i - \mathbf{c}_k\|^2,$$

Where $\mathbf{c}_k = \sum_{i=1}^{N} u_{ik}\mathbf{x}_i$ is the kth fuzzy Reassign patterns to clusters to re-duce this criterion function value and recompute U.

(3) Repeat step 2 until entries in U do not change significantly. In fuzzy clustering, each cluster is a fuzzy set of all the patterns. Figure 4 illustrates the idea. The rectangles en-close two “hard” clusters in the data: H1={1,2,3,4,5} and H2={6,7,8,9}. A fuzzy clustering algorithm might produce the two fuzzy clusters F1 and F2 depicted by ellipses. The patterns will have membership values in [0,1] for each cluster. For example, fuzzy cluster F1 could be compactly described as

{(1,0.9), (2,0.8), (3,0.7), (4,0.6), (5,0.55),(6,0.2), (7,0.2), (8,0.0), (9,0.0)},
and F2 could be described as
{(1,0.0), (2,0.0), (3,0.0), (4,0.1), (5,0.15), (6,0.4), (7,0.35), (8,1.0), (9,0.9)},

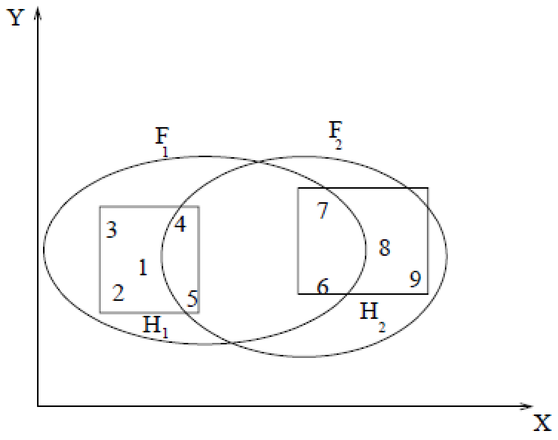

Figure 4. Fuzzy clusters.

The ordered pairs (i, μ_i) in each cluster represent the ith pattern and its membership value to the cluster μ_i. Larger membership values indicate higher confidence in the assignment of the pattern to the cluster. A hard clustering can be obtained from a fuzzy partition by thresholding the membership value.

2.4 Artificial Neural Networks for Clustering

Artificial neural networks (ANNs)] are motivated by biological neural networks. ANNs have been used extensively over the past three decades for both classification and clustering Some of the features of the ANNs that are important in pattern clustering are:

(1) ANNs process numerical vectors and so require patterns to be represented using quantitative features only.
(2) ANNs are inherently parallel and distributed processing architectures.
(3) ANNs may learn their interconnection weights adaptively . More specifically, they can act as pattern normalizers and feature selectors by appropriate selection of weights.

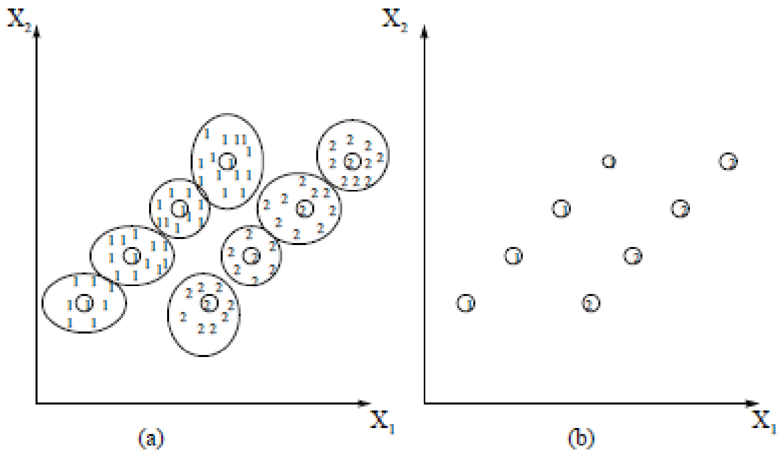

Figure 5. Data compression by clustering.

The architectures of these ANNs are simple: they are single-layered. Patterns are presented at the input and are associated with the out-put nodes. The weights between the in-put nodes and the output nodes are iteratively changed until a termination criterion is satisfied. Competitive learning has been found to exist in biological neural net-works. Further, its convergence is controlled by various parameters such as the learning rate and a neighborhood of the winning node in which learning takes place. It is possible that a particular input pattern can fire different output units at different iterations; this brings up the stability issue of learning systems

2.4 An Evolutionary Algorithm for Clustering

(1) Choose a random population of solutions. Each solution here corresponds to a valid k-partition of the data. Associate a fitness value with each solution. Typically, fitness is inversely proportional to the squared error value. A solution with a small squared error will have a larger fitness value.
(2) Use the evolutionary operators selection, recombination and mutation to generate the next population of solutions. Evaluate the fitness values of these solutions
(3) Repeat step 2 until some termination condition is satisfied

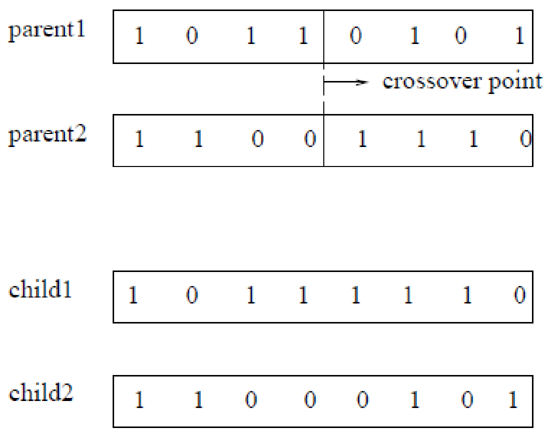

Figure 6. Crossover operation.

There are a variety of recombination operators in use; crossover is the most popular. Crossover takes as input a pair of chromosomes (called parents) and outputs a new pair of chromosomes (called children or offspring) as depicted in Figure 6. In Figure 6, a single point crossover operation is depicted. It exchanges the segments of the parents across a crossover point. For example, in Figure 6, the parents are the binary strings '10110101' and '11001110'. The segments in the two parents after the crossover point (between the fourth and fifth locations) are exchanged to pro-duce the child chromosomes. Mutation takes as input a chromosome and out-puts a chromosome by complementing the bit value at a randomly selected location in the input chromosome. For example, the string '11111110' is generated by applying the mutation operator to the second bit location in the string '10111110' (starting at the left). Both crossover and mutation are applied with some prespecified probabilities which depend on the fitness values.

2.6 Clustering Large Data Sets

There are several applications where it is necessary to cluster a large collection of patterns. The definition of 'large' has varied with changes in technology .In the 1960s, 'large' meant several thousand patterns; now, there are applications where millions of patterns of high dimensionality have to be clustered. For example, to segment an image of size 500 X 500 pixels, the number of pixels to be clustered is 250,000. In document retrieval and information filtering, mil-lions of patterns with a dimensionality of more than 100 have to be clustered to achieve data abstraction. A majority of the approaches and algorithms pro-posed in the literature cannot handle such large data sets. Approaches based on genetic algorithms, tabu search and simulated annealing are optimization techniques and are restricted to reason-ably small data sets. Implementations of conceptual clustering optimize some criterion functions and are typically computationally expensive. The convergent k-means algorithm and its ANN equivalent, the Kohonen net, have been used to cluster large data sets [Mao and Jain 1996]. The reasons behind the popularity of the k-means algorithm are:

(1) Its time complexity is $O(nkl)$, where n is the number of patterns, k is the number of clusters, and l is the number of iterations taken by the algorithm to converge. Typically, k and l are fixed in advance and so the algorithm has linear time complexity in the size of the data set [Day 1992]

(2) Its space complexity is $O(k + n)$ requires additional space to store the data matrix. It is possible to store the data matrix in a secondary memory and access each pattern based on need. However, this scheme requires a huge access time because of the iterative nature of the algorithm, and as a consequence processing time increases enormously.

(3) It is order-independent; for a given initial seed set of cluster centers, it generates the same partition of the data irrespective of the order in which the patterns are presented to the algorithm.

Table I. Complexity of Clustering Algorithms

Clustering Algorithm	Time Complexity	Space Complexity
leader	$O(kn)$	$O(k)$
k-means	$O(nkl)$	$O(k)$
ISODATA	$O(nkl)$	$O(k)$
shortest spanning path	$O(n^2)$	$O(n)$
single-line	$O(n^2 \log n)$	$O(n^2)$
complete-line	$O(n^2 \log n)$	$O(n^2)$

However, the k-means algorithm is sensitive to initial seed selection and even in the best case, it can produce only hyperspherical clusters.

Hierarchical algorithms are more versatile. But they have the following dis-advantages:

1) The time complexity of hierarchical agglomerative algorithms is $O(n \log^2 n)$ .It is possible to obtain single-link clusters using an MST of the data, which can be constructed in $O(n \log^2 n)$ time for two-dimensional data.
2) The space complexity of agglomerative algorithms is O(n2). This is be-cause a similarity matrix of size n X n has to be stored. To cluster every pixel in a 100 X 100 image, approximately 200 megabytes of storage would be required. It is possible to compute the entries of this matrix based on need instead of storing them (this would increase the algorithm's time complexity. .

Table I lists the time and space complexities of several well-known algorithms. Here, n is the number of patterns to be clustered, k is the number of terns to be clustered, k is the number of cluster .A possible solution to the problem of clustering large data sets while only marginally sacrificing the versatility of clusters is to implement more efficient variants of clustering algorithms. A hybrid approach was used in Ross [1968], where a set of reference points is chosen as in the k-means algorithm, and each of the remaining data points is assigned to one or more reference points or clusters. Minimal spanning trees (MST) are obtained for each group of points separately. These MSTs are merged to form an approximate global MST. This approach computes similarities between only a fraction of all possible pairs of points. It was shown that the number of similarities computed for 10,000 pat-terns using this approach is the same as the total number of pairs of points in a collection of 2,000 points. Bentley and Friedman [1978] contains an algorithm that can compute an approximate MST in $O(n \log^2 n)$ time. A scheme to generate an approximate dendrogram incrementally in $O(n \log^2 n)$ time was presented in Zupan [1982], while Venkateswarlu and Raju [1992] pro-posed an algorithm to speed up the ISO-DATA clustering algorithm. A study of the approximate single-linkage cluster analysis of large data sets was reported in Eddy et al. [1994]. In that work, an approximate MST was used to form single-link clusters of a data set of size 40,000.

The emerging discipline of data mining has spurred the development of new algorithms for clustering large data sets. Two algorithms of note are the CLARANS algorithm developed by Ng and Han [1994] and the BIRCH algorithm proposed by Zhang et al. [1996]. CLARANS (Clustering Large Applica-tions based on random Search) identifies candidate cluster centroids through analysis of repeated random samples from the original data. Because of the use of random sampling, the time complexity is O(n) for a pattern set of n elements. The BIRCH algorithm (Baanced Iterative Reducing and Clustering) stores summary information about candidate clusters in a dynamic tree data structure. This tree hierarchically organizes the clusterings represented at the leaf nodes. The tree can be rebuilt when a threshold specifying cluster size is updated manually, or when memory constraints force a change in this threshold. This algorithm, like CLAR-ANS, has a time complexity linear in the number of patterns.

The algorithms discussed above work on large data sets, where it is possible to accommodate the entire pattern set in the main memory. However, there are applications where the entire data set cannot be stored in the main memory because of its size. There are currently three possible approaches to solve this problem.

(1) The pattern set can be stored in a secondary memory and subsets of this data clustered independently, followed by a merging step to yield a clustering of the entire pattern set. We call this approach the divide and conquer approach.

(2) An incremental clustering algorithm can be employed. Here, the entire data matrix is stored in a secondary memory and data items are trans-ferred to the main memory one at a time for clustering. Only the cluster representations are stored in the main memory to alleviate the space limitations.

(3) A parallel implementation of a clustering algorithm may be used. We discuss these approaches in the next three subsections

2.7 Divide and Conquer Approach. Here, we store the entire pattern matrix of size n X d in a secondary storage space. We divide this data into p blocks, where an optimum value of p can be chosen based on the clustering algorithm used .Let us assume that we have n / p patterns in each of the blocks.

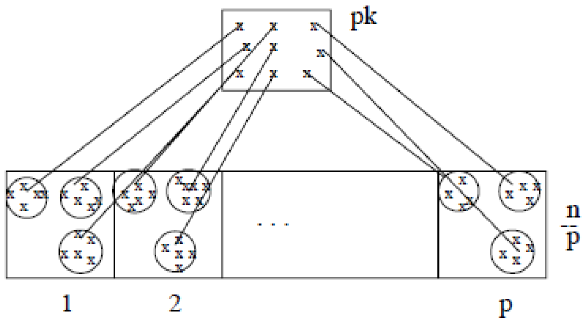

Figure 7. Divide and conquer approach to clustering.

We transfer each of these blocks to the main memory and cluster it into k clusters using a standard algorithm. One or more representative samples from each of these clusters are stored separately; we have pk of these representative pat-terns if we choose one representative per cluster. These pk representatives are further clustered into k clusters and the cluster labels of these representative patterns are used to relabel the original pattern matrix.. It is possible to extend this algorithm to any number of levels; more levels are required if the data set is very large and the main memory size is very small [Murty and Krishna 1980]. If the single-link algorithm is used to obtain 5 clusters, then there is a substantial savings in the number of computations as shown in Table II for optimally chosen p when the number of clusters is fixed at

A two-level strategy for clustering a data set containing 2,000 patterns was described in Stahl [1986]. In the first level, the data set is loosely clustered into a large number of clusters using the leader algorithm. Representatives from these clusters, one per cluster, are the input to the second level clustering, which is obtained using Ward's hierarchical method.

Table II. Number of Distance Computations (*n*) for the Single-Link Clustering Algorithm and a Two-Level Divide and Conquer Algorithm

n	Single-link	p	Two-level
100	4,950		1200
500	124,750	2	10,750
100	499,500	4	31,500
10,000	49,995,000	10	1,013,750

2.8 Incremental Clustering.

Incremental clustering is based on the assumption that it is possible to consider patterns one at a time and assign them to existing clusters. Here, a new data item is assigned to a cluster with-out affecting the existing clusters significantly. A high level description of a typical incremental clustering algorithm is given below.

An Incremental Clustering Algorithm

(1) Assign the first data item to a cluster.
(2) Consider the next data item. Either assign this item to one of the existing clusters or assign it to a new cluster. This assignment is done based on some criterion, e.g. the distance between the new item and the existing cluster centroids.
(3) Repeat step 2 till all the data items are clustered

The major advantage with the incremental clustering algorithms is that it is not necessary to store the entire pattern matrix in the memory. So, the space requirements of incremental algorithms are very small. Typically, they are noniterative. So their time requirements are also small. There are several incremental clustering algorithms:

(1) The leader clustering algorithm [Hartigan 1975] is the simplest in terms of time complexity which is $O(nk)$. It has gained popularity be-cause of its neural network implementation, the ART network [Carpenter and Grossberg 1990]. It is very easy to implement as it requires only $O(k)$ space.
(2) The shortest spanning path (SSP) algorithm [Slagle et al. 1975] was originally proposed for data reorganization and was successfully used in automatic auditing of records [Lee et al. 1978]. Here, SSP algorithm was used to cluster 2000 pat-terns using 18 features. These clusters are used to estimate missing feature values in data items and to identify erroneous feature values.
(3) The cobweb system [Fisher 1987] is an incremental conceptual cluster-ing algorithm. It has been successfully used in engineering applications [Fisher et al. 1993].
(4) An incremental clustering algorithm for dynamic information processing was presented in Can [1993]. The motivation behind this work is that, in dynamic databases, items might get added and deleted over time. These changes should be reflected in the partition generated without significantly affecting the current clusters. This algorithm was used to cluster incrementally an INSPEC database of 12,684 documents corresponding to computer science and electrical engineering.

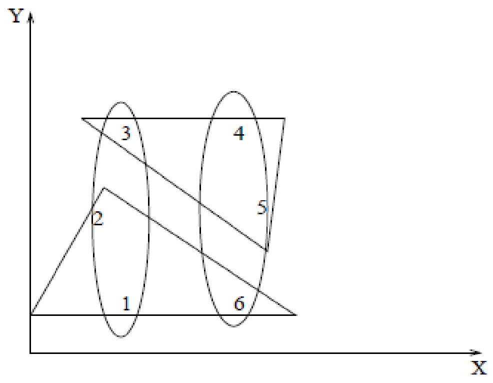

Figure 8. The leader algorithm is order dependent.

Order-independence is an important property of clustering algorithms. An algorithm is order-independent if it generates the same partition for any order in which the data is presented. Other-wise, it is order-dependent. Most of the incremental algorithms presented above are order-dependent. We illustrate this order-dependent property in Figure 8 where there are 6 two-dimensional objects labeled 1 to 6. If we present these patterns to the leader algorithm in the order 2,1,3,5,4,6 then the two clusters obtained are shown by ellipses. If the order is 1,2,6,4,5,3, then we get a two-partition as shown by the triangles. The SSP algorithm, cobweb, and the algorithm in Can [1993] are all order-dependent.

III. Conclusion

There are several applications where decision making and exploratory pattern analysis have to be performed on large data sets. It is possible to handle these problems if some useful abstraction of the data is obtained and is used in decision making, rather than directly using the entire data set. By data abstraction, we mean a simple and compact representation of the data. This simplicity helps the machine in efficient processing or a human in comprehending the structure in data easily. Clustering algorithms are ideally suited for achieving data abstraction.

In this paper, we have examined various steps in clustering: pattern representation, similarity computation, grouping process, and cluster representation. Also, we have discussed statistical, fuzzy, neural, evolutionary, and knowledge-based approaches to clustering.

Pattern recognition re-searchers conveniently avoid this step by assuming that the pattern representations are available as input to the clustering algorithm. In small size data sets, pattern representations can be obtained based on previous experience of the user with the problem. However, in the case of large data sets, it is difficult for the user to keep track of the importance of each feature in clustering. A solution is to make as many measurements on the patterns as possible and use them in pattern representation. But it is not possible to use a large collection of measurements directly in clustering because of computational costs.

A variety of schemes have been used to compute similarity between two patterns. They use knowledge either implicitly or explicitly. Most of the knowledge-based clustering algorithms use explicit knowledge in similarity computation. However, if patterns are not represented using proper features, then it is not possible to get a meaningful partition irrespective of the quality and quantity of knowledge used in similarity computation. There is no universally acceptable scheme for computing similarity between patterns represented using a mixture of both qualitative and quantitative features. Dissimilarity between a pair of patterns is represented using a distance measure that may or may not be a metric.

ANN-based clustering schemes are neural implementations of the clustering algorithms, and they share the undesired properties of these algorithms. However, ANNs have the capability to automatically normalize the data and extract features. An important observation is that even if a scheme can find the optimal solution to the squared error partitioning problem, it may still fall short of the requirements because of the possible non-isotropic nature of the clusters.

In some applications, for example in document retrieval, it may be useful to have a clustering that is not a partition. This means clusters are overlapping. Fuzzy clustering algorithms can handle mixed data types. However, a major problem with fuzzy clustering is that it is difficult to obtain the member-ship values. A general approach may not work because of the subjective nature of clustering. It is required to rep-resent clusters obtained in a suitable form to help the decision maker. Knowledge based clustering schemes generate intuitively appealing descriptions of clusters. They can be used even when the patterns are represented using a combination of qualitative and quantitative features, provided that knowledge linking a concept and the mixed features are available. However, implementations of the conceptual clustering schemes are computationally expensive and are not suitable for grouping large data sets.

The k-means algorithm and its neural implementation, the Kohonen net, are most successfully used on large data sets. This is because k-means algorithm is simple to implement and computationally attractive because of its linear time complexity. Divide and conquer is a heuristic that has been rightly exploited by computer algorithm designers to reduce computational costs. However, it should be judiciously used in clustering to achieve meaningful results.

In summary, clustering is an interesting, useful, and challenging problem. It has great potential in applications like object recognition, image segmentation, and information filtering and retrieval.

Acknowledgements

We would like to express our deep sense of gratitude to the organizing committee of DSMS Group of Institutions. Special thanks to all the laboratory staff of Bengal College Of Engineering for their noble help and cooperation to carry out our experiments.

References

[1] ALLEN, P. A. AND ALLEN, J. R. 1990. Basin Analysis: Principles and Applications. Blackwell Scientific Publications, Inc., Cambridge, MA

[2] BABU, G. P. AND MURTY, M. N. 1994. Clustering with evolution strategies. Pattern Recogn. 27, 321–329.

[3] BABU, G. P., MURTY, M. N., AND KEERTHI, S. S. 2000. Stochastic connectionist approach for pattern clustering (To appear). IEEE Trans. Syst. Man Cybern.

[4] BALL, G. H. AND HALL, D. J. 1965. ISODATA, a novel method of data analysis and classification. Tech. Rep.. Stanford University, Stanford, CA.

[5] BENTLEY, J. L. AND FRIEDMAN, J. H. 1978. Fast algorithms for constructing minimal spanning trees in coordinate spaces. IEEE Trans. Comput. C-27, 6 (June), 97–105.

[6] BEZDEK, J. C. 1981. Pattern Recognition With Fuzzy Objective Function Algorithms. Plenum Press, New York, NY.

[7] CARPENTER, G. AND GROSSBERG, S. 1990. ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. Neural Networks 3, 129–152.

[8] CHEKURI, C., GOLDWASSER, M. H., RAGHAVAN, P., AND UPFAL, E. 1997. Web search using automatic classification. In Proceedings of the Sixth International Conference on the World Wide Web (Santa Clara, CA, Apr.), http:// theory.stanford.edu/people/wass/publications/ Web Search/Web Search.html.

[9] CHENG, C. H. 1995. A branch-and-bound clustering algorithm. IEEE Trans. Syst. Man Cybern. 25, 895–898.

[10] CHENG, Y. AND FU, K. S. 1985. Conceptual clustering in knowledge organization. IEEE Trans. Pattern Anal. Mach. Intell. 7, 592–598.

[11] CHENG, Y. 1995. Mean shift, mode seeking, and clustering. IEEE Trans. Pattern Anal. Mach. Intell. 17, 7 (July), 790–799.

[12] DALE, M. B. 1985. On the comparison of conceptual clustering and numerical taxonomy. IEEE Trans. Pattern Anal. Mach. Intell. 7, 241–244.

[13] DAVE, R. N. 1992. Generalized fuzzy C-shells clustering and detection of circular and elliptic boundaries. Pattern Recogn. 25, 713–722.

[14] DAVIS, T., Ed. 1991. The Handbook of Genetic Algorithms. Van Nostrand Reinhold Co., New York, NY.

[15] DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. B. 39, 1, 1–38.

[16] EDDY, W. F., MOCKUS, A., AND OUE, S. 1996. Approximate single linkage cluster analysis of large data sets in high-dimensional spaces. Comput. Stat. Data Anal. 23, 1, 29–43.

[17] ETZIONI, O. 1996. The World-Wide Web: quagmire or gold mine? Commun. ACM 39, 11, 65–68.

[18] EVERITT, B. S. 1993. Cluster Analysis. Edward Arnold, Ltd., London, UK.

[19] FABER, V. 1994. Clustering and the continuous k-means algorithm. Los Alamos Science 22, 138–144.

[20] FABER, V., HOCHBERG, J. C., KELLY, P. M., THOMAS, T. R., AND WHITE, J. M. 1994. Concept extraction: A data-mining technique. Los Alamos Science 22, 122–149. FAYYAD, U. M. 1996. Data mining and knowledge discovery: Making sense out of data. IEEE Expert 11, 5 (Oct.), 20–25.

[21] GOWDA, K. C. AND DIDAY, E. 1992. Symbolic clustering using a new dissimilarity measure. IEEE Trans. Syst. Man Cybern. 22, 368–378.

[22] GOWER, J. C. AND ROSS, G. J. S. 1969. Minimum spanning rees and single-linkage cluster

[23] GREFENSTETTE, J 1986. Optimization of control parameters for genetic algorithms. IEEE Trans. Syst. Man Cybern. SMC-16, 1 (Jan./ Feb. 1986), 122–128.

[24] HARALICK, R. M. AND KELLY, G. L. 1969. Pattern recognition with measurement space and spatial clustering for multiple images. Proc. IEEE 57, 4, 654–665.

[25] HARTIGAN, J. A. 1975. Clustering Algorithms.John Wiley and Sons, Inc., New York, NY.

[26] HEDBERG, S. 1996. Searching for the mother lode: Tales of the first data miners. IEEE Expert 11, 5 (Oct.), 4–7.

[27] HOFFMAN, R. AND JAIN, A. K. 1987. Segmentation and classification of range images. IEEE Trans. Pattern Anal. Mach. Intell. PAMI-9, 5 (Sept. 1987), 608–620.

[28] HOFMANN, T. AND BUHMANN, J. 1997. Pairwise data clustering by deterministic annealing. IEEE Trans. Pattern Anal. Mach. Intell. 19, 1 (Jan.), 1–14.

[29] HOFMANN, T., PUZICHA, J., AND BUCHMANN, J. M. 1998. Unsupervised texture segmentation in a deterministic annealing framework. IEEE Trans. Pattern Anal. Mach. Intell. 20, 8, 803–818.